AFIT/GSO/ENG/95M-01

ACQUIRING CONSISTENT KNOWLEDGE FOR BAYESIAN FORESTS

THESIS

Darwyn O. Banks, Captain, USAF

AFIT/GSO/ENG/95M-01

AFIT/GSO/ENG/95M-01

ACQUIRING CONSISTENT KNOWLEDGE FOR BAYESIAN FORESTS

THESIS

Presented to the Faculty of the Graduate School of Engineering

of the Air Force Institute of Technology

Air University

in Partial Fulfillment of the

Requirements for the Degrees of

Master of Science in Space Operations

Master of Science in Computer Science

Darwyn O. Banks, B.S.

Captain, USAF

March 1995

Approved for public release; distribution unlimited

Acknowledgements

"I can do all things through Christ which strengtheneth me."
— Philippians 4:13

My committee:	Dr. E. Santos, Jr.; Col. G. Parnell, Ph.D. and Dr. B. Woodruff.
Doc:	For all his assistance and insight even if I didn't always catch the clue train. For coding up the prototype inference engines used in this research, and more importantly, not making me do it. For all the long nights talking through problems fifteen ways to Sunday while we debugged my code and his. For interesting mealtimes—"Cigarette butts in gallstone sauce"—and for introduction to the finer points of Anime.
Maj Woodruff (ret):	For coming back out of retirement to continue his involvement with my work.
NASA LeRC:	June, Claudia, Tim, Bill, and Chris for their time, willingness to help, and most importantly, for supplying the HPOTP application to me and helping me learn it.
GSO-94D:	For being exactly the kind of wild and crazy (the faculty might go so far as to say, unruly) bunch of guys and gals that could keep my sanity level high enough (and my stress level low enough) actually to earn 2 degrees.
Maj Gunsch:	For taking such a strong interest in my research and the input thus provided.
Dr. Hartrum:	For being a willing victim of the numerous drive-by Zed attacks until I accepted the correctness of Appendix A.
LTC Lawlis (ret), Maj Bailor:	The past and present chiefs of ENG's Computer Science and Engineering Division for all their assistance and support in my hare-brained effort to go "over and above" to meet the GCS program requirements.
Santos' guinea pigs:	The original PESKI-teers—Chuck, Eric, Long and I; Ed, the doctoral student; Shawn, Duck & Jerry—the next generation.

This work was sponsored in part by AFOSR Grant #94-0006.

Table of Contents

List of Figures

AFIT/GSO/ENG/95M-01

Abstract

This thesis develops a methodology and a tool for knowledge acquisition with the new probabilistic knowledge representation—the Bayesian Forest. It establishes the structure of the Knowledge Acquisition and Maintenance module of the Probabilities, Expert Systems, Knowledge and Inference (PESKI) architecture. The tool, MACK, is designed to be used directly by the domain expert(s) rather than by knowledge engineer(s), and thus supports automated knowledge acquisition.

This research determines and implements the constraints necessary to ensure the consistency of Bayesian Forest knowledge bases as data is both acquired and subsequently maintained. The impact to the PESKI architecture of time-dependent information and default assumptions during reasoning is also explored. The tool has been applied to NASA's Post-Test Diagnostics System which locates anomalies aboard the Space Shuttles' Main Engines.

1. INTRODUCTION

Automated reasoning draws much of its potency from the way in which its knowledge is organized and stored. These knowledge representations are the underlying data structures in electronic memory which can be accessed for use by the computer. They determine the methods of computational inferencing available to the expert system and, by extension, the range of problems to which automated reasoning may be applied. The problems of acquiring knowledge for the representation can make the task of building a useful and reliable expert system difficult indeed.

. . . Flexible Knowledge Representation

There is an inverse relationship between the efficiency of an automated reasoning algorithm and the flexibility of its associated knowledge representation. However, the former is secondary to the overall capabilities of the final product when weighed against the latter. Without an appropriate representation we cannot properly store our knowledge. Thus, system designers often give preference to flexible representations [18]. Nevertheless, real-world applications for intelligent or expert systems, such as those in space operations domains, require a balance of these two capabilities. As we shall see, Bayesian Forests [49]—an extension of the more common Bayesian networks—are just such a representation which can provide both a flexible knowledge structure and efficient inferencing algorithms over that representation.

The more flexible a knowledge representation is, the more successfully it can model the given domain. It must "bend" itself to accommodate the uncertainties common in life and other real-world applications. For example, many would agree with the general "rule of thumb" that "All birds can fly." However, such rules alone are insufficient. Every rule has its exceptions. Penguins, ostriches and the television character Big Bird, to name a few, cannot fly. Thus, the statement "All birds can fly" is neither completely true nor completely false. Its distance from these extremes is its uncertainty. While we can reduce that uncertainty by the explicit incorporation of exceptions into the rule, *i.e.*, "All birds, other than Big Bird, can fly unless they are penguins or ostriches," the sheer number of exceptions often makes this prohibitive. In our avian example new exceptions might include turkeys, emus, hatchlings and any bird with a broken wing, for starters.

Inference mechanisms such as those in Bayesian Forests are noteworthy for their ability to represent such uncertainty precisely because they marry the strong models of probability theory with an "if-then" rule structure similar to those rules which Shortliffe and Buchanan [62] found most natural for human experts. Reliance on the well-established laws of probability helps guarantee that Bayesian Forests are a sound and consistent representation of knowledge [53], [49], and therefore, that the results they generate will not be inconsistent.

. . . Forests versus Networks

The line between Bayesian Forests and Bayesian networks begins with notions of flexibility, as previously discussed. It continues into the methods available to each model

for efficient computation. Computing with Bayesian networks is NP-hard [12], [60], however, forests build upon algorithmic approaches from the field of operations research which demonstrate polynomial growth rates in the expected case [53], [49]. Moreover, these algorithms can be specialized for various domains.

At the lowest level the primary distinction between the two Bayesian reasoning representations—Bayesian Forests and Bayesian networks—is in their respective treatment of the random variables over which they are defined. In either reasoning scheme these variables represent objects or events.[1] The assignment of a value to a random variable instantiates it, and a complete set of instantiations, *i.e.*, the assignment of one value to each and every random variable, defines what we call a state of the world [51], [52], [54].

These differences between Bayesian networks and Forests are evident in two areas:

1) The topological ordering of the *random variables*, and

2) The definition of the *conditional independencies*.

There is a particular topological ordering of the random variables of the Bayesian network and these random variables define the conditional independencies within the network. On the other hand, the Bayesian Forest maintains a topological ordering of the *instantiations* of its random variables. Similarly, the instantiations define the forest's conditional independencies.

In other words, the random variable is the Bayesian network's smallest component for reasoning purposes. This combined with the new conditional independencies also

[1]For our purposes, we also assume them to be discrete.

frees the forest from the necessity common to other Bayesian analysis systems like Bayesian networks for a **complete** specification of all possible probabilities [40]. The Bayesian Forest inference engine can function with incomplete information.

For example, given an automobile and a traffic signal as Bayesian network random variables and some value, z_i, between 0 and 1, Bayesian network construction allows us either to predicate the automobile's motion upon the state of the signal,[2]

$$P(\text{ car} = \text{stopped} \mid \text{signal} = \text{red }) = z_1 \tag{1-1}$$

or the current state of the signal upon the auto,

$$P(\text{ signal} = \text{green} \mid \text{car} = \text{on\_detector }) = z_2 \tag{1-2}$$

but not both as this would preclude any topological ordering over these two variables.[3] In contrast, the topological ordering implicit in a Bayesian Forest is based not upon random variables alone, but upon the various instantiations that are available for those variables. Thus, a Bayesian Forest's topological ordering would prevent the introduction of equation (1-3) into the knowledge base because of its potential conflict with equation (1-1) while permitting us to include both equations (1-1) and (1-2)—the apparently cyclic example above—precisely because the assignments to the variables differ [49].

[2]Read: "The probability that the car is stopped *given* that the light is red." "Car = stopped" is the *head* of the probability; "light = red," its *tail*. See Chapter 2 for further details.

[3]This topological ordering determines the conditional independencies between random variables and becomes an important factor in the eventual calculation and storage of probabilities.

$$P(\text{ car} = \text{stopped} \mid \text{signal} = \text{red }) = z_1 \tag{1-1}$$

$$P(\text{ signal} = \text{red} \mid \text{car} = \text{stopped }) = z_3 \tag{1-3}$$

. . . Knowledge Acquisition

Because Bayesian Forests are a relatively new representation, very little work has been done to date on methods of acquiring knowledge with them. Forests provide a more powerful knowledge representation [53], [49], but it remains to be seen if this new representation better lends itself to new or existing methods of knowledge acquisition.

Feigenbaum [18] describes the acquisition of new knowledge as an aspect of an automated reasoning system which eclipses both its knowledge representation and its inferencing capabilities in importance. Knowledge acquisition's central role in artificial intelligence derives from the fact that "knowledge is power" [10]. Simply stated, the performance of an expert system is a direct function of both the quality and quantity of the encoded knowledge. Thus, in general, more knowledge enhances the performance of our intelligent systems. However, it must be more of the right kind of knowledge, organized and codified in ways applicable and appropriate to our symbolic computations.

The critical subareas of an expert system include: its reasoning methodologies, the organization of its knowledge base, the validity of that knowledge, and of course, its interface with the user (See Figure 1.1) [53], [27], [70], [28]. As we can see simply from their titles, knowledge is integral to two of these four components. As such, the details of knowledge acquisition—especially when that knowledge includes uncertainty, *i.e.*, incomplete information and/or facts which are neither entirely true nor entirely false—can

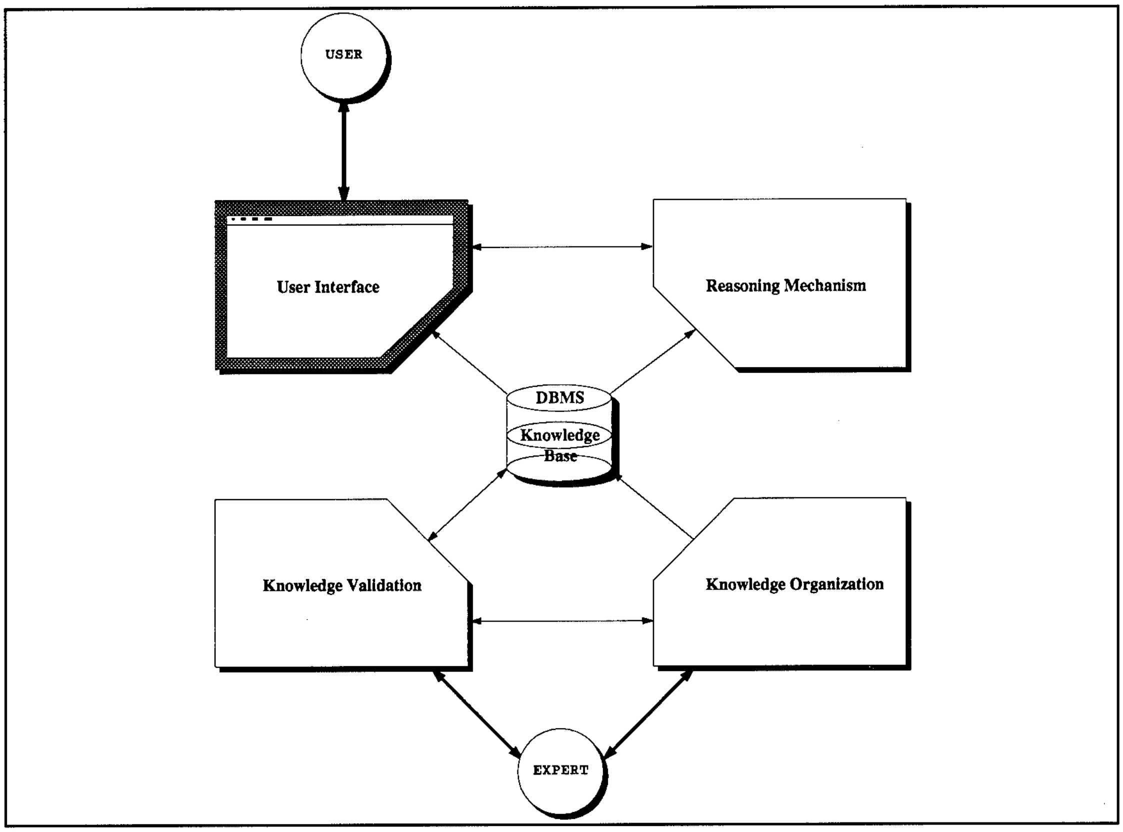

Figure 1.1 The major subcomponents of an expert system [53]

become major impediments to the development of real-world expert systems [53].

. . . Knowledge Acquisition by Articulation

Currently, knowledge acquisition methods fall into one of three categories: articulation, induction and automated [13]. Articulation can be described as the "person-to-person" approach to knowledge acquisition. It requires the services of a knowledge engineer to interview the expert about her/his area(s) of expertise, to decide upon an appropriate knowledge representation for the data, to encode the information, and finally to validate the accuracy of the final product. Often, articulative methods try the expert's patience since they include the engineer shadowing the expert in and around the

workplace constantly inquiring about the underlying reasons of this or that action [27], [70], [28].

Obviously, the success of articulation methods depends heavily not only upon the skills of the knowledge engineer, but equally so upon the openness of the expert and his or her ability to communicate knowledge to the engineer. Much research has been done on methodologies to improve this process [57], [20], [19], [37]. Ideally, the knowledge engineer is a good student—at least of the subject at hand—and the expert is a comparable teacher of the same. In effect, the engineer is apprenticing him- or her-self to the expert for a period of time sufficient only to glean the basics and transcribe them into machine language. In reality, though, true apprenticeships involve years of study and practice. While it is still the most common form of knowledge acquisition, articulation is both a time- and labor-intensive process that iterates as the knowledge engineer regularly returns to the expert for verification [13], [27].

. . . Knowledge Acquisition by Induction

Knowledge acquisition by induction [67] requires the expert to enumerate the members of a set of random variables, now called attributes, and their associated values. Then, for all possible assignments of values to this set, s/he asserts a hypothesis or conclusion for each member attribute. This type of knowledge acquisition is the basis for Quinlan's ID3 knowledge acquisition algorithm [11]. Inductive assumptions presume the given hypotheses to be mutually exclusive and correct, all the attributes relevant and all attribute values discrete. Unfortunately, inductive acquisition techniques such as ID3 pay

a very high price in that they require complete test cases to stave off degradations in rule-set confidence. Clearly, as the numbers of attributes and values grow, the size of the question set presented to the expert quickly becomes combinatorially large [13].

For example, a set of eight attributes, four of which have three possible values, and two pairs of which have four and five, yields 32,400 ($3^4 \times 4^2 \times 5^2$) value combinations each of which requires eight hypotheses or conclusions from the expert (one for each attribute). Few experts can be expected to want or to be able to contribute over a quarter of a million inputs of unknown complexity for such a small system! Larger systems are clearly prohibitive.

Moreover, inductive knowledge acquisition approaches like the ID3 algorithm develop rule decision trees using entropy functions. Such trees may require the inference engine to analyze the entire structure for a single hypothesis, much less for the entire set. Their required structure builds redundancy into the knowledge base of a system which is already costly [13].

. . . Automated Knowledge Acquisition

The last approach, automated knowledge acquisition, makes an effort completely to dispense with the knowledge engineer. In actuality the knowledge engineering tasks have been transferred elsewhere in the software development process such that the program itself can query the expert and format the information received. This method identifies and partitions knowledge into modules that are tailored to a particular knowledge representation or expert system shell. The use of modules constrains and

guides the expert through the process by receiving input, checking the consistency of the data already entered, and querying the expert to correct any inconsistencies found [13], [69], [35].

. . . MACK

Our work here is in support of a new, integrated expert system framework called PESKI—PROBABILITIES, EXPERT SYSTEMS, KNOWLEDGE AND INFERENCE. PESKI's overall goal seeks strong semantics and more effective algorithms for reasoning with uncertainty [53]. Specifically, this paper addresses Knowledge Acquisition and Maintenance, one of

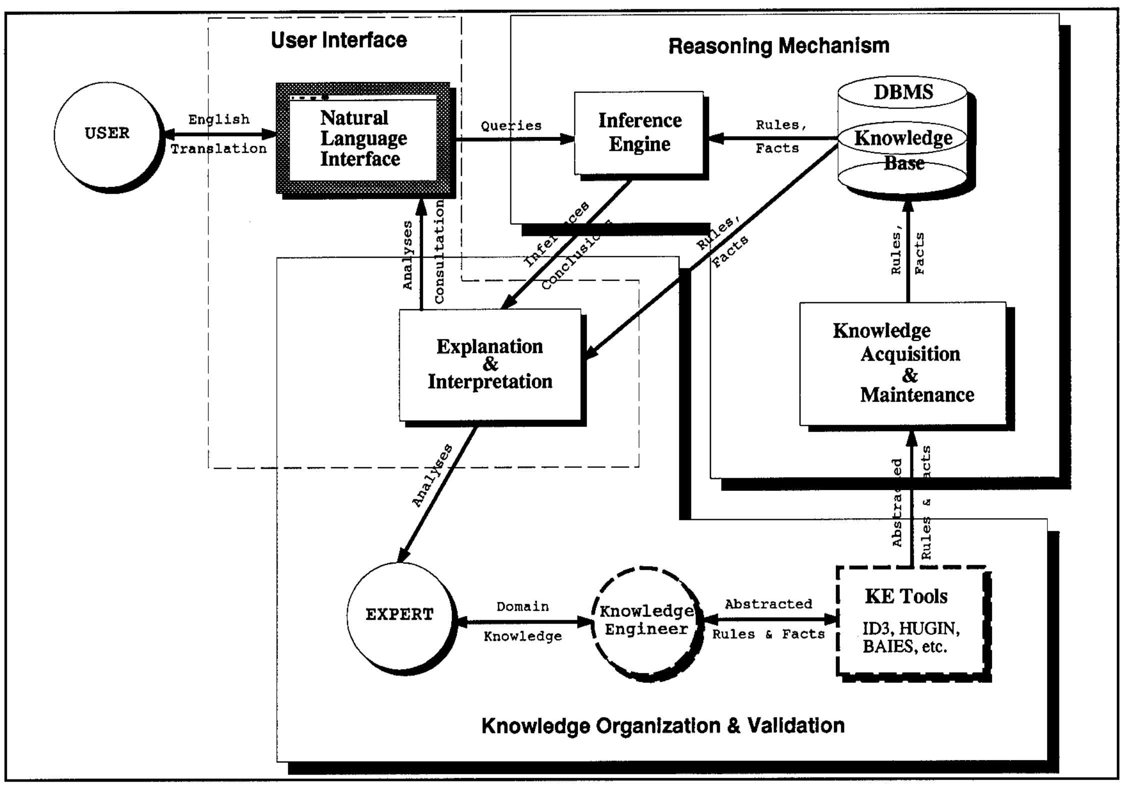

Figure 1.2 The PESKI Architecture [53]

PESKI's four major components (See Figure 1.2[4]). The others—the Inference Engine, the Explanation & Interpretation facilities, and the Natural Language Interface—are being or will be explored separately (See [4], [56]). Because of the manner in which we match the application to a method, our approach to the development of a knowledge acquisition tool for PESKI and Bayesian Forests can be termed "middle-out."[5]

In this paper we present an automated knowledge acquisition tool for Bayesian Forests called MACK, the Module for the Acquisition of Consistent Knowledge. MACK is a tool designed not only to acquire, but also to maintain, knowledge bases while guaranteeing the consistency of the knowledge therein.

By combining some of the strengths of inductive and automated knowledge acquisition methods, MACK enhances the development of probabilistic reasoning systems like PESKI while also addressing issues of temporal logic and default reasoning which often arise. Individually, each of these areas—probabilistic reasoning, temporal reasoning, and reasoning with default assumptions—has been shown to be NP-hard [12], [60], [55], [49]. This work determines their intersection with the new PESKI expert system architecture and integrates those aspects, as appropriate, into the Knowledge Acquisition

---

[4]Broken-bordered objects in this figure, specifically Knowledge Engineer and Knowledge Engineering Tools, are considered optional in this architecture [53].

[5]The "middle-out" approach develops knowledge acquisition techniques for a method (in this case, Bayesian Forests), then applies them to one or more particular domains. "Bottom-up" tool-building works from a specific problem domain and incorporates much of the domain knowledge into the resultant tool. The "top-down" uses general analysis and minimal knowledge of the domain [8], [9], [25], [14], [36], [17], [38].

and Maintenance module, instantiated here with the Bayesian Forest knowledge representation.

Presently, researchers at the National Aeronautics and Space Administration (NASA) are developing an intelligent system to facilitate post-test diagnosis of the main engines of the Space Transportation System, more commonly known as the Space Shuttle [7], [43], [48]. Knowledge acquisition by articulation for this system has been in progress for over thirty months, yet the net result to date is a single complete component which reasons over data from the shuttle's high pressure oxidizer turbopump. We apply our efforts to this part of the NASA project.

No previous work on the Post-Test Diagnostic System (PTDS) has taken a probabilistic approach to the problem.[6] In fact, system designers rejected both rule-based and probabilistic methodologies in favor of a case-based look-up scheme for the systems module. They opted against the former because of causal feedback loops in the knowledge, while the use of probabilities was estimated to add an order of magnitude to problem complexity [5]. Bayesian Forests, however, provide the tools to handle the causal loops and the flexibility better to balance the strictures of probabilistic reasoning with the uncertainties common to a real-world domain like PTDS.[7] It has been our intention to demonstrate these capabilities of Bayesian Forests via the re-development of

[6]Real-time sensor data validation modules do use Bayesian information fusion techniques. However, the sensor validation system, which has indirect control of a manned vehicle, is separate from the solely advisory Post-Test Diagnostics [6], [73].

[7]Previous work on the particular PTDS component we chose presently handles uncertainty only by prepending the words: "possible," "may indicate," or "may be" to 91.6% of non-historical anomaly reports.

the high-pressure oxidizer turbopump component using the MACK tool. Chapter 2 explains the concerns we faced specific to Bayesian Forests and how automatically to identify inconsistencies found in Bayesian Forest knowledge bases. Chapter 3 describes the Space Shuttle application itself, while Chapter 4 addresses procedural issues of knowledge acquisition using MACK.

2. BAYESIAN FORESTRY

Up to this point we have alluded to Bayesian Forests without a proper description. Bayesian Forests, the most recent manifestation of Santos' Bayesian Multinetworks [49],[8] are a rule-based probabilistic reasoning methodology. Probabilistic reasoning in intelligent systems exploits the fact that probability theory is and has been an accepted language both for the description of uncertainty and for making inferences from incomplete knowledge [33]. Using the semantics of probability theory, we designate random variables to represent the discrete objects or events in question. We then assign a joint probability distribution to each possible state of the world, *i.e.*, a specific value assignment for each random variable. This assignment allows us to reason over the set of probabilities [2], [3].

Unfortunately, the size of this joint distribution can grow exponentially in the number of random variables making solutions and knowledge acquisition computationally intractable [33], [42]. One way to address this complexity is by assuming many, if not all, of the random variables to be independent. However, while such independence assumptions significantly facilitate knowledge acquisition and ultimate resolution, if applied carelessly, they can oversimplify the problem statement such that the final answer loses considerable validity [33].

[8]Originally dubbed "Bayesian Multi-Nets," the name has been changed to "Bayesian Forests" to distinguish them from the Bayesian multinetworks of Heckerman and Geiger [22].

Bayesian systems' avoid oversimplification by couching their independence assumptions in terms of conditional dependencies. Let D, E and F be random variables. The conditional probability, $P(D|E,F)$, identifies the belief in D's truth given that E and F are both known to be true. If $P(D|E,F) = P(D|E)$, we say that random variables D and F are conditionally independent given E. In other words, once we know E is true, we can establish D's truth with or without any knowledge of F's [41].

Bayesian philosophy holds that such conditional relationships—*e.g.*, $P(D|E)$—are more in keeping with the way humans tend to organize knowledge [31], [59], [58]. Equation (2-1) below shows Bayes' Formula for computing these probabilities. We can view

$$P(D|E) = \frac{P(E,D)}{P(E)} \tag{2-1}$$

random variable D as a possible hypothesis (or set of hypotheses) held in advance and E as the actual evidence that was or will be generated. The formula shows how previous hypotheses should be modified in light of that new evidence [45].

$$P(E,D) = P(D|E)\,P(E) \tag{2-2}$$

Equation (2-2) manipulates Bayes' Formula to allow us to compute the joint distribution. It generalizes to n variables as shown in Equation (2-3)[9]. More importantly to

$$P(D,E,F,G,H) = \begin{array}{r} P(D|E,F,G,H)\;P(E|F,G,H) \\ P(G|F,H)\;P(H|F)\;P(F) \end{array} \tag{2-3}$$

[9]Equation (2-3) shows one of the n! possible expansions. n is obviously equal to 5 in this example.

PESKI's inferencing, the subsequent incorporation of the known independence conditions further reduces the amount of information we must actually store to be able to compute the required joint probability [41], [39], [33], [52], [51], [54]. Thus, let us assume that random variable E is known to be conditionally independent from both F and H when the value of G is known, and D is likewise conditionally independent with knowledge of both E and G. Then, we can simplify Equation (2-4) further still (see Equation (2-3)).

$$P(D,E,F,G,H) = P(D|E,G)\ P(E|G)\ P(G|F,H)\ P(H|F)\ P(F) \qquad (2\text{-}4)$$

These conditional dependencies can also be represented pictorially with a directed graph. Furthermore, the graph must be acyclic for a Bayesian network, else it would be impossible to determine an equation for computing the joint distribution similar to equation (2-4) using the conditional dependencies.

We illustrate Bayesian Forests with just such a graphical comparison to the more conventional Bayesian network. Let A, B and C be random variables in a Bayesian network representing a traffic light, its associated vehicle detector and pedestrian signal, respectively. Figure 2.1 graphically depicts this network over these variables.

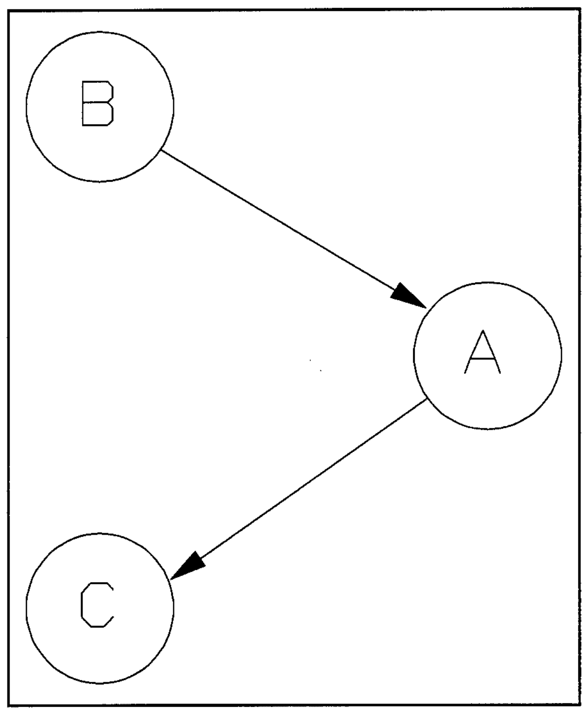

Figure 2.1 Example Bayesian Network Graph

Since the signal depends upon the light, we say that A is the parent of C. Similarly, B is the parent of A.

Now, assume we want for some reason to expand this set with a probability for the detector stating the likelihood of its being tripped during rush hour. Such an inclusion would introduce a cycle into our Bayesian network since the detector and traffic light cannot both depend upon the other. It becomes synonymous to the classic circular reasoning example: "If Smoke, then Fire" coupled with "If Fire, then Smoke."

Herein lies the added flexibility of the Bayesian Forest. Assuming the same trio of random variables and the partial set of values below:

$$P(C = \text{"Don't Walk"} \mid A = \text{red}) = x_1 \tag{2-5}$$

$$P(C = \text{"Walk"} \mid A = \text{green}) = x_2 \tag{2-6}$$

$$P(A = \text{green} \mid B = \text{On}) = x_3 \tag{2-7}$$

$$P(A = \text{red} \mid B = \text{Off}) = x_4 \tag{2-8}$$

We can quite legally add the new constraint:

$$P(B = \text{On} \mid A = \text{red}, D = \text{rush hour}) = x_5 \tag{2-9}$$

without creating a directed cycle. Figure 2.2 shows the graphical representation of this example.

Notice how the graph of a Bayesian Forest is a simple, bipartite and directed.[10] It has two distinct types of nodes. The first, shown as lettered ovals, corresponds to nodes found in a Bayesian network. However, now these nodes are particular not simply

---

[10]See [71] for further discussion of graph definitions.

to a random variable, but to a specific instantiation thereof. Hence Figure 2.1's single node for variable A, the traffic light, becomes two distinct instantiation nodes for "A = Red" and "A = Green" in Figure 2.2.

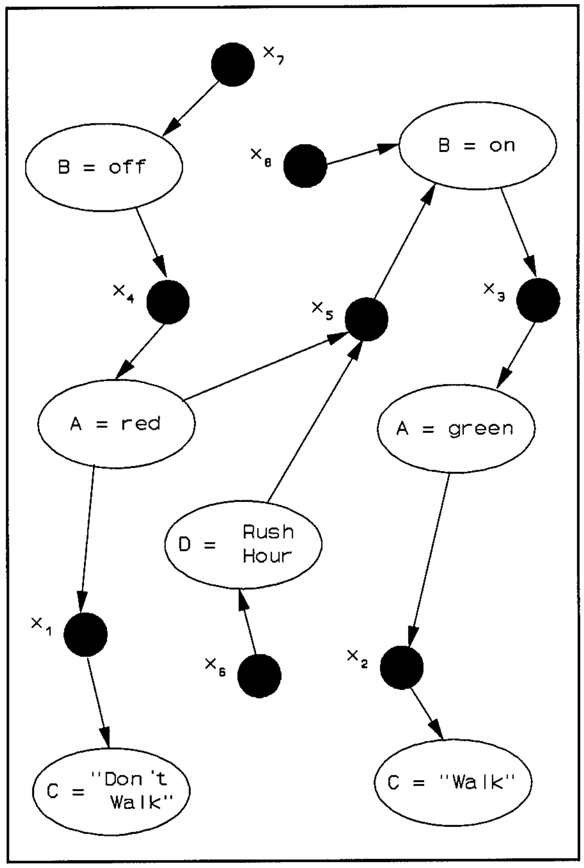

Figure 2.2 Example Bayesian Forest Graph

The second type of node, depicted as a blackened circle, is called a support node. When drawn, these nodes, which represent the numeric probability value itself, have exactly one outbound arrow to the instantiation node representing the probability's head. Support nodes also provide a graphical terminus for zero or more inbound dependency or conditioning arrows from each of the parent instantiations in the probability's tail. Conversely, all instantiations nodes must have one or more inbound arrows to establish their probabilities and sets of supporting conditions, but need not contain any outbound arrows.

Reasoning algorithms for Bayesian Forests are built upon inference engines previously developed for weighted, "AND/OR" directed acyclic graphs (WAODAG) [49], [4], [41], [39]. In these schema Bayesian Forest support nodes correspond to WAODAG AND nodes precisely because all parent instantiation nodes must be active or true before the support may be. Likewise, a Bayesian Forest's instantiation nodes correspond roughly

to the WAODAG's OR node since any one of an instantiation node's support conditions is sufficient to activate it. The correspondence is inexact in that the Bayesian Forest instantiation node actually represents an exclusive-or condition. Any support condition may substantiate an instantiation node, but that support node must be the **only** one active.

As we have seen, Bayesian Forests allow ambidirectional construction, *i.e.*, we can have $P(A=a_1 \mid B=b_2, C=c_1)$ and $P(B=b_2 \mid A=a_2)$ in our database simultaneously. However, this additional constructive capacity changes the concept of consistency for a knowledge base in ways unique to Bayesian Forests.

Bayesian Forest inference algorithms operate straightforwardly by computing a joint probability for a particular assignment of values to every variable available in the random variable set—henceforth we will refer to such an assignment as a "state of the world." In an inconsistent forest there will be multiple ways to compute this value for any particular state of the world with no guarantee that the results will be equal as they must [49].

To illustrate, let X, Y and Z be boolean random variables. Clearly, there are eight possible states of the world. Given the following probabilities as the entire population of the forest's database:

$$P(X = \text{true} \mid Y = \text{false}) = .40 \tag{2-10}$$
$$P(X = \text{true} \mid Z = \text{true}) = .80 \tag{2-11}$$
$$P(Y = \text{false} \mid X = \text{true}) = .70 \tag{2-12}$$
$$P(Y = \text{false} \mid Z = \text{true}) = .45 \tag{2-13}$$
$$P(Z = \text{true}) = .75 \tag{2-14}$$

The inference engine may compute $P(X = \text{true}, Y = \text{false}, Z = \text{true})$[11] by multiplying equations (2-10), (2-13) and (2-14) or by multiplying equations (2-12), (2-11) and (2-14). However, the joint probability on the first of these logical paths is .135, while along the second path it is .42, more than three times greater!

In order to develop a functional inference engine, we eliminate this inconsistency by requiring any group of compatible[12] probabilities which share a head to have exactly equal probability values. In other words, we guarantee that the probabilistic value of equation (2-11) is the same as the value of equation (2-10) and that equation (2-13) is also equal to (2-12) as shown below.[13]

$$P(X = \text{true} \mid Y = \text{false}) = .40 \tag{2-10a}$$
$$P(X = \text{true} \mid Z = \text{true}) = \cancel{.80}\ .40 \tag{2-11a}$$
$$P(Y = \text{false} \mid X = \text{true}) = .70 \tag{2-12a}$$
$$P(Y = \text{false} \mid Z = \text{true}) = \cancel{.45}\ .70 \tag{2-13a}$$
$$P(Z = \text{true}) = .75 \tag{2-14a}$$

While this equality requirement clearly forbids inconsistencies, it does little either to explain or to assist the knowledge engineer in his or her efforts to build the system. The engineer's problem, then, is to determine the equating formula which will be used

---

[11]Note that there is insufficient data here to compute any other joint probability.

[12]We define probabilities to be *incompatible* or mutually exclusive only if there exists a random variable in the tail (see footnote 2) of both which takes on a different value in each.

$$P(Y = \text{false} \mid Z = \text{true}) = .45$$
$$P(Y = \text{false} \mid Z = \text{false}) = .83$$

For example, these probability equations are incompatible since both place conditions upon random variable Y's being false and the variable Z assumes a different value in their tails. Similarly conditioned probabilities with non-identical heads are considered *mutually exclusive.*

[13]In this case we have arbitrarily set the value of the second member of each pair equal to the first.

initially to create a viable knowledge base. Such formulae are countless—*e.g.*, minima, maxima, weighted or unweighted averages, any real function over the values, etc. Moreover, in the absence of other information, all can be equally valid. This makes an algorithm to construct Bayesian Forests all the more elusive.

Before we can develop a knowledge acquisition methodology, we must be aware of those areas of a Bayesian Forest with the potential to violate its construction constraints or to harbor inconsistent bits of knowledge. Once identified, we can ensure both the validity and consistency of a forest by induction as it is being built. However, unlike inductive systems predicated on the ID3 algorithm, Bayesian Forests have no requirement for the complete specifications of all attributes and values which make those systems less tenable for large data sets.

Following are descriptions of the eight construction constraints we determined, as well as the manner in which our implementation satisfies them. Taken together, they guarantee both the structure of the Bayesian Forest and, more importantly, its probabilistic validity. The fact that there are only eight underscores the flexibility of the Bayesian Forest representation and its ability to obey the laws of probability theory while still being general enough directly to interface with the expert. Appendix A is an object-oriented object model of a Bayesian Forest using *Z* notation [26], [66], [46]. The model gives a formal mathematical specification of each constraint.

- **Constraint 1**: This first constraint recognizes the fact that a Bayesian Forest stores all its probabilistic information in its support nodes,

rather than its instantiation nodes. Thus, we restate here the requirement that all instantiation nodes in a forest be the head of, at least, one support node.

- **Constraint 2**: Because we implement instantiation nodes and support nodes as separate but interrelated object classes, this constraint ensures that the instantiations referenced by any support node, *i.e.*, the sources and sinks of its arrows in the graph, be well-defined.

- **Constraint 3**: Here we simply guarantee that all instantiation nodes taken together form a set, not a family [71], *i.e.*, contain no duplicate instantiations. Different instantiations of the same item must have distinct values.

- **Constraint 4**: This fourth constraint encapsulates Santos' original requirement that any support nodes which share a head instantiation must be mutually exclusive [49]. Given any state of the world, all but one of an instantiation node's support conditions must conflict with that particular assignment of global values. In the parlance of logic, we could say constraint 4 requires the truth or falsity of any instantiation node to be established via an exclusive-or condition among its attendant support nodes.

$$P(X = \text{true} \mid Y = \text{true}) = y_1 \tag{2-15}$$

$$P(X = \text{true} \mid Y = \text{true}, Z = \text{false}) = y_2 \tag{2-16}$$

$$P(X = \text{true} \mid Y = \text{false}, Z = \text{false}) = y_3 \tag{2-17}$$

$$P(X = \text{true} \mid Y = \text{false}, Z = \text{true}) = y_4 \tag{2-18}$$

For example, the equations above show a set of support conditions using boolean random variables: X, Y and Z. Clearly, each of these support conditions modifies the same head instantiation: setting X to "true." However, probabilities (2-15) and (2-16) are not mutually exclusive, since the former, which does not depend upon random variable Z, will always be valid any time that the latter is. In this case when Y is true, either probability affords a valid inference path to substantiate X's truth, thus y_1 must equal y_2 for the database to be consistent.

Constraint 1 guarantees there will be one or more support nodes for each instantiation. This fourth constraint provides the necessary distinctions between those support nodes such that one, and only one, may be active.

- **Constraint 5**: The fifth constraint closes the door on logical cycles. Given a particular inference chain, it prevents the reoccurrence of a support node's head in the tails of its successors in that chain. For example, equations (2-19) through (2-22) below form a loop in that all four can be

$$P(A = a_1 \mid D = d_2) \tag{2-19}$$
$$P(D = d_2 \mid B = b_3, C = c_1) \tag{2-20}$$
$$P(B = b_3 \mid E = e_2) \tag{2-21}$$
$$P(E = e_2 \mid A = a_1, C = c_1) \tag{2-22}$$

simultaneously active, *i.e.*, no mutual exclusivities exist within the set, and equation (2-22) depends in part upon the head instantiation of a predecessor,

in this case "$A = a_1$." Failure to preclude this cycle would allow the inference engine potentially to enter an infinite loop since equation (2-19) can clearly be re-investigated as a successor to (2-22).

As with many search problems, discovering these cycles can quickly become combinatorial. However, we can conduct this search *as the expert identifies support conditions* which ensures the consistency of the knowledge base. This also assists the expert in correcting inconsistencies by flagging them sooner, *i.e.*, upon introduction to the database, rather than later. We accomplish this search cheaply and efficiently using a depth-first algorithm which begins at the head of the new support node and branches throughout the Bayesian Forest structure, as necessary.

o **Constraint 6**: Because we are concerned only with one particular state of the world at a time, it is obviously unacceptable to have one instantiation of an item depend upon a different instantiation of that same item since both can never be true simultaneously. In fact, the probability shown in equation (2-23) and all others like it would always have to equal zero, per force.

$$P(Y = \text{true} \mid Y = \text{false}, Z = \text{true}) \tag{2-23}$$

Moreover, it is even less acceptable for the veracity of an instantiation to be dependent upon the instantiation itself as in equation (2-24).

$$P(Y = \text{true} \mid X = \text{false}, Y = \text{true}) \qquad (2\text{-}24)$$

o **Constraint 7**: Following similar logic as constraint 6, one item cannot be conditioned by multiple values of another precisely because the instantiations of the second item obviously contradict themselves. Thus, equation (2-25) is clearly invalid since the random variable X cannot both be true *and* false at the same time.

$$P(Y = \text{true} \mid X = \text{false}, Z = \text{true}, X = \text{true}) \qquad (2\text{-}25)$$

o **Constraint 8**: Because we are using a probabilistic reasoning scheme, we need this last constraint to disallow any simultaneously valid sets of probabilities for the same item from ever summing to values greater than 1. We use the fact that the head instantiations of the sets' elements share the same random variable to identify each set.

In the example below, we have collected all support conditions for random variable A (regardless of instantiated value) which depend upon random variable B's first value.

$$P(A = a_1 \mid B = b_1, C = c_1) = v \qquad (2\text{-}26)$$
$$P(A = a_1 \mid B = b_1, C = c_2, D = d_1) = w \qquad (2\text{-}27)$$
$$P(A = a_2 \mid B = b_1) = x \qquad (2\text{-}28)$$
$$P(A = a_3 \mid B = b_1, D = d_1) = y \qquad (2\text{-}29)$$
$$P(A = a_3 \mid B = b_1, D = d_2) = z \qquad (2\text{-}30)$$

Equations (2-26) and (2-27) can never be active at the same time since they depend on different instantiations of C. Similarly, equation (2-30) is mutually exclusive both with equation (2-27) and equation (2-29) due to random variable D. Under constraint 8 these dependencies on different states of the world divide this set of equations such that the following inequalities must all be true:

$$x + z \leq 1 \tag{2-31}$$

$$v + x + z \leq 1 \tag{2-32}$$

$$w + x + y \leq 1 \tag{2-33}$$

$$v + x + y \leq 1 \tag{2-34}$$

MACK automatically normalizes the values of any probabilities which violate this constraint simply by dividing each element of the set by the total.[14] Notice that equation (2-28)'s probability, x, is a factor in all these subsets since it does not depend either on C or D and can thus be simultaneously true with any of the other four. We also note that in this example equation (2-32) effectively overrides (2-31) because if the former holds, then the latter must also *a fortiori*.[15]

A complete consistency check essentially involves verification of each of the eight aforementioned constraints. Obviously, such an approach may become computationally

[14]The pre-normalized value is maintained in the system for use in subsequent normalizations over different sets of probabilities.

[15]v, w, x, y, z are all non-negative real-valued variables between 0 and 1.

expensive, especially as the size of the network grows. However, we have implemented our Bayesian Forest knowledge acquisition routine such that complete consistency checking is rarely necessary. Specifically, guarantors for constraints 2, 3, 6 and 7 are built into the object creation routines. Thus, as the expert defines the items of the forest, their associated values and dependencies, the software objects themselves prohibit duplicate instantiations and ensure the validity of all references between the instantiation node and support node classes. Constraints 1, 4, 5 and 8, then, are the only constraints explicitly tested during a review. In addition, our Bayesian Forest implementation conducts such reviews incrementally. We check each new support node as it is entered into the database to ensure it introduces no new inconsistencies to the existing consistent forest, *e.g.*, by engendering a logical cycle.

We now turn our attention to an actual application of MACK. The application comes from the National Aeronautics and Space Administration (NASA). The next chapter discusses NASA's Post-Test Diagnostic System for the Space Shuttle's main engines and how the MACK knowledge acquisition tool performed in that domain.

3. THE POST-TEST DIAGNOSTIC SYSTEM

Development, maintenance and improvement of any large system, especially one with human lives at stake, usually involves extensive testing. NASA's Space Transportation System, or Space Shuttle, is no exception. Marshall Space Flight Center in Huntsville, Alabama routinely conducts ground tests and collects actual flight data on the shuttle's boosters better to assess the health, status and current capabilities of the reusable engines and their many components. Presently, these assessments involve large teams of engineers who review remote data received from hundreds of on-board sensors called PIDs. Officials then use these manual reviews to determine the fitness of the engine for another test or flight [7].

The Post-Test Diagnostic System is an on-going cooperative project to automate the Space Shuttle Main Engine (SSME) review process using intelligent systems. Its stated goals are [73]:

- to aid in the detection and diagnosis of engine anomalies
- to increase accuracy and repeatability of rocket engine data analysis
- to reduce analysis time

When complete, its components will validate engine sensors, reliably extract salient features from telemetry data, and analyze SSME performance systems, combustion devices, turbomachinery and dynamics.

As of this writing, two different versions of one component—the High Pressure Oxidizer Turbopump (HPOTP)—have been built and validated by government contractors under the auspices of researchers at NASA Lewis Research Center in Cleveland, Ohio.[16] These systems provided us the opportunity to test MACK's applicability to a real-world domain and a set of known parameters against which to corroborate the utility of MACK-acquired knowledge for Bayesian Forest reasoning.

The HPOTP is an engine component designed initially to raise, then to maintain the pressure of the liquid oxygen flowing into the engine at the varying levels of thrust during the shuttle's flight profile [64]. Using a turbine powered by the oxidizer preburner's hydrogen-rich hot gas, this centrifugal pump manages the flow of liquid oxygen into the engine's main and preburner injectors. Beside the pumps and turbines, the HPOTP's third major group of subcomponents contains the extensive shaft seals which separate pumps, turbines and the fluids they regulate [64].

Being an automated tool, MACK is designed to be operated directly by the domain expert. In fact, it is a key component of the PESKI Knowledge Organization and Validation subsystem which considers a human knowledge engineer optional [53]. As a result, we, the knowledge engineers, simulated the expert's involvement.[17] We note, however, that much of the previous knowledge engineering accomplished for HPOTP has involved collating and sorting the information gathered in numerous interviews with the

[16]The first was developed by Science Applications International Corporation, San Diego, California [43]. The second by personnel from Aerojet Propulsion Division, Sacramento, California [7].

[17]Future work with MACK will involve its direct use by experts to develop a new expert module.

Alabamian crew of rocket scientists.[18] Appendix B correlates selected text from knowledge acquisition interviews with anomaly definitions from the second version of HPOTP and with conditional probabilities in the HPOTP Bayesian Forest. These correlations show that it is, in fact, plausible partially or completely to remove the middleman and allow the expert to be his/her own knowledge engineer—*i.e.*, if the expert is so inclined, s/he can with minimal instruction create a Bayesian Forest from scratch.

Appendix C contains transcripts of a knowledge acquisition session used to create part of a Bayesian Forest for HPOTP. Session data were taken from interview transcripts and reflect some prior knowledge engineering. We also presume to reason over the salient information gathered from the raw data by the existing feature extractor [43].

[18]NASA Lewis researchers have been conducting interviews with Marshall Space Flight Center engineers since Spring 1992.

4. KNOWLEDGE PROCESSING WITH MACK

Having described both Bayesian Forests and the HPOTP application, we now explore some of the processes by which MACK acquires knowledge. The PESKI architecture assumes the knowledge engineer to be optional (see Figure 1.2) [53]. As a result, the MACK tool, like those discussed by Sandahl [47], is intended to be the primary interface with the expert. This role places a premium on user-friendliness as much as adherence to Bayesian Forest constructs and probabilistic formalisms.

MACK is a menu-driven system. These menus allow us to handle the simpler Bayesian Forest constraints—constraints #2, 3, 6 & 7 (see Chapter 2)—by simple manipulation of the menu options presented to the user. Other illegal choices simply trap program control until a valid selection is entered. The examples excerpted below are taken from the HPOTP application.[19] It includes data entry of the conditions governing the shift anomaly noted via sensor, PID 990, here called "Anomaly 990 Shift." This anomaly depends upon the sensor's peak and equilibrium values which are represented by the random variables, "PID 990 Peak" and "PID 990 Equilibrium," respectively. Here the expert is creating the first support condition for the instantiation of the random variable "Anomaly 990 Shift" to value "Found."[20]

[19] See Appendix C for the entire transcript.

[20] Anomaly variables are basically boolean: "Found" or "Not Found."

At present, Anomaly 990 Shift's being Found depends upon the following
sets of conditions:

No support conditions!

Enter 0 to add new support conditions for
Anomaly 990 Shift's being Found
Otherwise, enter 1 to quit
0

Recall that Constraint 2 requires the instantiations and supports connected in a Bayesian Forest to be well-defined. Thus the system when creating a support condition only presents a choice among the previously instantiated random variables. MACK additionally restricts the options to those variables which can actually be used in the nascent support condition. We can see this in the absence of Anomaly 990 Shift itself from the subsequent menu shown below which is in keeping with Constraint 6.

Anomaly 990 Shift's being Found
can depend upon which of the following components:

2 -- PID 990 Equilibrium
3 -- PID 990 Peak
0 -- None of the Above Components

Choice: 2

1 -- Nominal
2 -- Out of Family
0 -- None of the Above; Abort

Choice: 2

Having already selected a value of PID 990 Equilibrium, the expert is now queried for continuance. In this abbreviated example we see that the only remaining random variable option available to the expert is PID 990 Peak. Anomaly 990 Shift has been

previously removed under Constraint 6 and PID 990 Equilibrium, as a new addition to the condition, is now illegal in accordance with Constraint 7.

Presently, this condition holds that Anomaly 990 Shift's being Found can depend upon the following:

PID 990 Equilibrium = Out of Family

Do you wish to extend this condition? Y / N _y_

Anomaly 990 Shift's being Found
can depend upon which of the following components:

3 -- PID 990 Peak

Choice: _3_

1 -- Nominal
2 -- Out of Family -- High
3 -- Out of Family -- Low
0 -- None of the Above; Abort

Choice: _2_

Presently, this condition holds that Anomaly 990 Shift's being Found can depend upon the following:

PID 990 Equilibrium = Out of Family
PID 990 Peak = Out of Family -- High

Do you wish to extend this condition? Y / N _n_

Please complete the sentence below from the following list of choices:

0 -- inconceivable
1 -- not likely
2 -- possible
3 -- probable
4 -- almost certain

It is ____________ that the Anomaly 990 Shift is Found depending upon . . .

PID 990 Equilibrium = Out of Family
PID 990 Peak = Out of Family – High

Choice: _3_

MACK then presents the expert with a menu of choices from which it will internally derive the support condition's probability. Since a Bayesian Forest's probabilistic nature is masked from the expert, we use only the qualitative and linguistic terms shown below with their current value ranges.[21]

inconceivable:	0.00 - 0.10
not likely:	0.10 - 0.35
possible:	0.35 - 0.65
probable:	0.65 - 0.90
almost certain:	0.90 - 1.00

It is important to note here that during knowledge acquisition for a Bayesian Forest, the actual numeric value assigned to any given probabilities is not significant. Refinement of these values through belief revision and belief updating is the province of the forest's reasoning and explanation facilities.[22] The values associated with each node only attain meaning after the inference engine reasons over them during belief updating. It should be obvious, however, the inference engine's propagation of probabilities must begin somewhere. In his discussion of the validity of such values to probabilistic reasoning schemes, Pearl [41] writes:

> [p. 148, The] conditional probabilities characterizing the links in the network do not seem to impose definitive constraints on the probabilities that can be assigned to the nodes. . . . The result is that any arbitrary assignment of beliefs to the propositions *a* & *b* can be consistent with the value of P(*a*|*b*) that was initially assigned to the link connecting them . . . .

[21]Other research in this area has shown the commonality of adjective connotations. Current work seeks to refine these further [23], [16], [34], [2], [3], [15], [24], [65].

[22]Future research efforts will focus on implementations of Bayesian Forest inference mechanisms [53], [4], [56], [52], [51], [54].

Thus, the decision to use a random number generator in the initial stages of database development neither adds to nor detracts from the Bayesian Forest. More germane to the topic at hand, it certainly has no impact upon the consistency of the data's logical organization within that forest.

These menu restrictions only account for the simple constraints. The more involved Bayesian Forest formalisms are found in a separate consistency checking routine. MACK initiates this larger routine itself after any change to the set of support conditions, removal of an instantiation or upon receipt of up to five new, unsupported instantiations.

```
Welcome to M.A.C.K. --   the Bayesian Forest Module for the
                         Acquisition of Consistent Knowledge!!

     0 - Generate new Bayesian Forest
     1 - Edit existing Bayesian Forest
     2 - Display current Bayesian Forest
     3 - Load Bayesian Forest from file
     4 - Save Bayesian Forest to file
     5 - Check Forest Consistency
     6 - Run Bayesian Forest Belief Revision Program
     7 - Delete the current Bayesian Forest
     8 - Exit Bayesian Forest program
```

This consistency checking routine sampled below covers the four remaining Bayesian Forest constraints and, as a courtesy, also notifies the user of any conditions with zero probability. Initially, we see below that the forest has failed Constraint 1 since the system has no condition defining a probability value for the absence of Anomaly 990 Shift. In these cases, MACK prompts the expert appropriately. Were the expert to answer any of these negatively, the consistency routine aborts there and returns the expert to the edit menu.

This Bayesian Forest is currently inconsistent.
Is it correct that
Anomaly 990 Shift being Not Found
does not depend on anything else? Y/N _y_

Please complete the sentence below from the following list of choices:

0 -- inconceivable
1 -- not likely
2 -- possible
3 -- probable
4 -- almost certain

It is ____________ that the Anomaly 990 Shift is Not Found depending upon . . .
Nothing!

Choice: _3_

This Bayesian Forest is currently inconsistent.
Is it correct that
PID 990 Equilibrium being Nominal
does not depend on anything else? Y/N _n_

Please edit the conditions for
PID 990 Equilibrium being Nominal accordingly.

Growing trees for your Bayesian Forest.
Instantiations:
0 - Add new instantiation
1 - Delete instantiation

Support Conditions:
2 - Add new support condition
3 - Edit existing support condition
4 - Delete support condition

5 - Return to main menu

Constraint 4 is an important one which identifies support conditions that are not mutually exclusive. With insufficient information to make any automatic resolution assumptions here, MACK again queries the expert.

ERROR: Support conditions below are not mutually exclusive.

At present, Anomaly 990 Shift's being Found depends upon the following sets of conditions:

Support Node #1:
  PID 990 Equilibrium = Out of Family
  PID 990 Peak = Out of Family -- High
Support Node #2:
  PID 990 Peak = Out of Family -- Low
  PID 990 Equilibrium = Nominal
Support Node #3:
  PID 990 Equilibrium = Nominal

This Bayesian Forest is currently inconsistent.
The following pair of conditions for Anomaly 990 Shift being Found are not mutually exclusive.

First Set:
  PID 990 Peak = Out of Family -- Low
  PID 990 Equilibrium = Nominal
Second Set:
  PID 990 Equilibrium = Nominal

Does Anomaly 990 Shift's being Found
  really depend upon both sets of conditions? [Enter 0]
    or
  upon each set separately? [Enter 1]

Choice: 1

Which of these conditions may we add to eliminate the overlap?
  1 -- PID 990 Peak can be Nominal
  2 -- PID 990 Peak can be Out of Family -- High
  0 -- None of the Above

Choice: 2

The expert is given the option either of merging the two conditions into one or of distinguishing them in some way. While the first option is straightforward,[23] the second could conceivably draw upon any component in the forest except the one in question, *i.e.*,

[23]This merger cannot be illegal, *i.e.*, violate either Constraint 6 or 7. If the support conditions in question reference instantiations which will conflict when merged, then they are mutually exclusive and, therefore, not inconsistent.

the head of the two support conditions. To assist the expert in this area, MACK makes an initial simplifying assumption that excludes all random variables that are not already present. Since the only way to establish mutual exclusion is for both of the support conditions to contain in their tails a different instantiation of one or more variables (see Chapter 2). The basis of the assumption is that at least one of the current random variables can be expanded to meet this requirement, thus allowing the tool automatically to select and present options as it does in other areas. These options will be all the values of the existing variables which are not already represented. MACK can easily determine which of the two support nodes should obtain the adjustment since, of course, Constraint 7 which proscribes against multiple values remains in effect.

Verifications of Constraint 8, shown below, occur somewhat innocuously. Since the expert is not aware of the actual probabilistic values anyway, MACK can simply normalize the pertinent sums[24] and reports any adjustments of the support conditions' qualitative variables—*e.g.*, inconceivable, not likely, possible, probable, or almost certain—to the expert. These normalizations always use the original probabilistic range the expert assigned in order to avoid a new, high-value addition from overwhelming predecessors whose values may have already been reduced.

```
This Bayesian Forest is inconsistent.
Currently, support ranges overlap. Adjusting ranges for consistency . . .
Conditions were:
```

[24]It is worth noting that although the normalization itself may be a trivial operation, the determination of the support node sets which are eligible to be normalized is not. Constraint 8's multi-tiered mathematical representation in Appendix A is a testament thereto.

It is probable that the Anomaly 990 Shift @ T_i is Found depending upon
. . .
PID 990 Equilibrium @ T_i = Out of Family
PID 990 Peak @ T_i = Out of Family -- High

It is probable that the Anomaly 990 Shift @ T_i is Not Found depending upon . . .
Nothing!

New conditions are:

It is possible that the Anomaly 990 Shift @ T_i is Found depending upon
. . .
PID 990 Equilibrium @ T_i = Out of Family
PID 990 Peak @ T_i = Out of Family -- High

It is possible that the Anomaly 990 Shift @ T_i is Not Found depending upon . . .
Nothing!

The HPOTP application turned out to be a rather flat forest. By that we mean that the sensor readings which represent the bulk of the random variables are unconditioned, and most anomaly determinations depend directly on the sensors rather than on some intermediate calculations. As a result, the application did not violate Constraint 5 which searches for logical cycles in the knowledge base.

Acquiring Temporal Information

Many real-world domains require the capability to model knowledge that changes over time. This requirement is even more pronounced in the PTDS domain which NASA eventually hopes to operate in real-time [5]. For a knowledge acquisition tool such as MACK, this introduces new developmental difficulties. While we want the interface with the expert to remain responsive and user-friendly, we must of course maintain the formalities of the system's probabilistic requirements.

Temporal aspects of HPOTP are best illustrated by the fact that the sensor readings are anything but static. During any given test, a sensor may take on multiple values, *e.g.*, Nominal, Erratic, Spiked, etc. Clearly, this violates the Bayesian Forest's requirement that variable instantiations be unique and mutually exclusive. However, by partitioning the test period into time slices we can accommodate these changes.

For purposes of MACK's knowledge acquisition, the tool queries the expert for the number of changes in a particular temporal variable to expect at run-time. It then uses the largest of these when creating the Bayesian Forest.[25] Appendix D demonstrates our method of parsing these changes into the timeline. Presently, MACK arbitrarily limits a support node's relative time dependencies to the current time period and either adjacent period—*i.e.*, the ones immediately before or after T_i[26]—however, expansion of this range to ± n intervals, if necessary, is very straightforward.

The MACK interface's approach to the time-dependencies within a domain follow the similar protocols to those outlined for the constraints above. The distinctions reside in the way the tool queries the expert for the temporal dependency of each new random variable and the special menu options which accommodate those variables so identified.

Please enter the name of the new component:
Item Name: PID 990 Peak

PID 990 Peak is a new component.
New INODE created: PID 990 Peak = No value given!!

[25] We have limited ourselves to 5 or fewer time intervals since the prototype reasoner used in support of this research is not powerful enough to handle the numbers of temporal random variables generated.

[26] In keeping with the system's intended flexibility, these qualitative temporal choices are "naturally linguistic" based on Allen's results [55], [1].

REMINDER:

Bayesian Forest Variables are persistent and mutually exclusive.
In other words, they take on 1, and only 1, of their possible values.
Obviously, this will not accommodate variables that change over time.

Can the value of PID 990 Peak vary with time?
Y / N _y_

How many times might PID 990 Peak change? _5_

When these temporal components are encountered during forest construction, MACK again queries the expert for the appropriate relative time dependencies of the support condition. We have now included temporal conditions into our previous example shown below. The "< UNDEFINED >" linguistic variable is a placeholder pending the expert's assignment of a legitimate probabilistic range after entering all the tail values.

At present, Anomaly 990 Shift's being Found depends upon the following sets of conditions:

Support Node #1:
PID 990 Equilibrium = Out of Family
PID 990 Peak = Out of Family -- High

Enter 0 to add new support conditions for
Anomaly 990 Shift's being Found
Otherwise, enter 1 to quit
_0_

Anomaly 990 Shift's being Found
can depend upon which of the following components:

2 -- PID 990 Equilibrium
3 -- PID 990 Peak
0 -- None of the Above Components

Choice: _3_

1 -- Nominal
2 -- Out of Family -- High
3 -- Out of Family -- Low
0 -- None of the Above; Abort

Choice: 3

It is < UNDEFINED > that the Anomaly 990 Shift @ T_i is Found depending upon . . .

PID 990 Peak @ T_i = Out of Family -- Low

New addition, PID 990 Peak, is time-dependent.
We should read its value, Out of Family -- Low, from which time interval?

-1 -- Time period immediately preceding Anomaly 990 Shift = Found
0 -- The same time period as Anomaly 990 Shift = Found
1 -- Time period immediately after Anomaly 990 Shift = Found

0

Presently, this condition holds that Anomaly 990 Shift's being Found can depend upon the following:

PID 990 Peak = Out of Family -- Low

Do you wish to extend this condition? Y / N y

Anomaly 990 Shift's being Found
can depend upon which of the following components:

2 -- PID 990 Equilibrium

Choice: 2

1 -- Nominal
2 -- Out of Family
0 -- None of the Above; Abort

Choice: 1

It is < UNDEFINED > that the Anomaly 990 Shift @ T_i is Found depending upon . . .

PID 990 Peak @ T_i = Out of Family -- Low
PID 990 Equilibrium @ T_i = Nominal

New addition, PID 990 Equilibrium, is time-dependent.
We should read its value, Nominal, from which time interval?

-1 -- Time period immediately preceding Anomaly 990 Shift = Found
0 -- The same time period as Anomaly 990 Shift = Found
1 -- Time period immediately after Anomaly 990 Shift = Found

-1

Presently, this condition holds that Anomaly 990 Shift's being Found can depend upon the following:

PID 990 Peak @ T_i = Out of Family -- Low
PID 990 Equilibrium @ T_i-1 = Nominal

Do you wish to extend this condition? Y / N n

Please complete the sentence below from the following list of choices:

0 -- inconceivable
1 -- not likely
2 -- possible
3 -- probable
4 -- almost certain

It is ____________ that the Anomaly 990 Shift @ T_i is Found depending upon . . .

PID 990 Peak @ T_i = Out of Family -- Low
PID 990 Equilibrium @ T_i-1 = Nominal

Reasoning with Defaults

Default values for a Bayesian Forest's random variables become important in cases where the system has incomplete information in the knowledge base with which to reason. These can include instances where the expert excludes pertinent data points, intentionally or otherwise, or instances wherein his/her implicit assumptions about the domain are not explicitly entered into the program.[27]

[27]The former type of incomplete information represents necessary knowledge which the expert neglected to incorporate, while the latter is information which by virtue of expertise s/he considers self-evident. In either case if the missing data causes the inference engine to assume a value, the system will highlight the deficiency to the expert.

When completed, the PESKI architecture will handle these situations via the Explanation and Interpretation Facility (See Figure 1.2) which will present the system's output to the human expert to ensure the correctness both of the inference engine's results and the logical choices made to arrive at those results. If the value of a random variable is assumed in order to reach a solution, that variable must be flagged to the user. In the absence of essential information, these explanations will very likely contain assumptions of which the expert may approve or disapprove. In either case they will highlight the system's lack of information prompting the expert for its inclusion.

However, PESKI is still under development as evidenced by our own work here on its knowledge acquisition facility. In the interim we have developed a prototype inference engine on which to test and validate MACK's acquired knowledge. In addition, this reasoner exercises the tool's ability to manipulate default data in an incomplete knowledge base while maintaining the required constraints of probability theory. We accomplish this by modelling incomplete information about a random variable with probabilistic sums that are strictly less than 1.[28] There are two very distinct approaches, then, by which to account for the deficit [40]: we can define default values as supersets of the given support conditions or as the negation of those same rules.

The superset approach absorbs the probability values of existing rules, if any, for the default instantiation into a larger set which includes them and all the undefined probability values, *i.e.*, that fraction which brings the overall sum up to 1. In this case,

[28]As we have seen in Chapter 2, the sum of probabilities for compatible instantiations must not exceed 1. However, Bayesian Forests have no requirement for the support conditions' sums always to equal 1 either.

we assume the default to be the most common occurrence and the rules to be identified exceptions thereto. "Default as negation" instead of adjusting the support conditions entered by the expert, simply creates an assumption node with a probability value which completely or partially complements the others, *i.e.*, its addition to the set maintains a sum less than or equal to 1.[29] Unlike the superset default, this assumption node can be activated if and only if all other support conditions fail. Now, the default—*i.e.*, the assumption node—has become the exception and the given rules constitute the common cases.

For an example let's return to the simple traffic light. Suppose each of the signal's three states—red (the default state), yellow and green—each has a probability of 0.3 with the remaining 0.1 unassigned. Superset default logic would absorb the uncounted 0.1 into the default's value such that red's value increases to 0.4. However, it masks the original, defined condition for the light's being red which may contain important information about the "red" state. The traffic light is simply assumed to be red unless the conditions for yellow or green are satisfied.

Negation, on the other hand, does not change any of the known data. Instead, it places part of the 0.1 value into an Assumption node. Then, if and only if no existing support condition for the traffic signal can be satisfied **and** the signal's value is pertinent to the solution being generated will the inference engine assume a value for the light.

[29] The alternative to this assumption node is completely to specify all the possible instantiations that have not yet been enumerated and then assign explicit values to each. Needless to say, this solution paves the way for a combinatorial explosion in the required number of such instantiations, many of which do not serve to enhance the knowledge base since we can assume the expert would have included them if they did.

Notice that this assumed value need not be red. It will instead be the value that best supports the current solution, red or otherwise.

In a probabilistic system the primary concern for default reasoning is the continued adherence to the laws of probability. Clearly, the superset approach does this, however, in the process it can also give too much credence to its default value, and possibly remove seminal knowledge established by the expert. The resultant system runs the risks of choosing the default value more often than is warranted solely because of its artificially higher probability. Likewise, "default as negation" also maintains probabilistic validity, but without the artificial inflation of values as the assumption node is kept separate from all others. Its value may increase, but only with the express approval of the expert. Nevertheless, this latter approach can also become preferential to its previous assumptions in those cases where the defined probabilities all have small values.

PESKI is designed to take the negation approach, thus MACK (and its proto-reasoner) support this position. Initially, we defend against overreliance upon the default by guaranteeing the probability of the assumption node to be one or more orders of magnitude lower than any legitimate assignment.[30] However, during feedback the expert may give her/his permission to incorporate an assumption as an actual default support condition in its own right. In this case, we increase its value according to the probability

[30]The prototype inference engine operates using integer linear programming and a variant of the simplex algorithm [72], [30]. In this model the assumption node always has the extreme value, M—"Big M" [52]. As a result, any calculable state of the world which does not use this value will always produce a better solution.

range the expert assigns and continue to reason normally.[31] Constraint 8 (see Chapter 2) will then ensure the probabilities' sum is less than 1. Also, since the head of this default support condition can duplicate an existing instantiation but will not have a tail of its own, we must also bypass Constraint 4's requirement for uniqueness. Specifically, a default support node is, by definition, mutually exclusive with the set of all others, thus it must, of course, be incompatible with any individual member of that set.

In diagnostic domains such as HPOTP, we are further protected from overreliance on a default since the sensor readings are given for any test. Deduction on these readings will either support the possibility of an anomaly as defined by the experts [7], [48] or not. Assumptions of default values are much more common in abductive systems [50] which would receive as input the presence of an anomaly and then would attempt to posit the most likely set of evidence, in this case sensor readings, which might have caused that anomaly to occur.

We choose here an example of this default dilemma from the PTDS domain. Sensors #327 and #328 measure pressure levels on the turbopump's balance piston. They can report any of the following conditions: Nominal, Level Shift ±, or Spike ±.[32] The difference between these two sensors' readings is itself meaningful: "Delta Level Shift 327-328." This difference can be "Positive," "Negative" or "Zero" where a non-zero value indicates a change in the sensors' relative values and "Zero" represents no such

[31]Although termed the exception, the probability value of the default support condition could, in fact, be quite high relative to the its sibling nodes. To the inference engine there is no distinguishing characteristic of a default support node. It is a value like any other available for computation.

[32]See Appendix D for graphical example of these values from the sensor data.

change. Input data from the experts include the following four support conditions governing "Delta Level Shift 327-328" [7], [43]:

It is probable that Delta Level Shift 327-328 is Positive depending upon . . .

PID 327 = Level Shift +
PID 328 = Level Shift -

It is probable that Delta Level Shift 327-328 is Negative depending upon . . .

PID 327 = Level Shift -
PID 328 = Level Shift +

It is probable that Delta Level Shift 327-328 is Zero depending upon . . .

PID 327 = Level Shift -
PID 328 = Level Shift -

It is probable that Delta Level Shift 327-328 is Zero depending upon . . .

PID 327 = Level Shift +
PID 328 = Level Shift +

This random variable highlights the difficulties of reasoning with defaults as there exist no explicit support conditions to define Delta Level Shift's value under any other circumstances, *e.g.*, those when either sensor is spiking up or down or both are. It is precisely this situation which occurs as the reasoner attempts to determine the likelihood of "Anomaly 5.06.1".[33] A "Spike" value in either sensor and "Zero" for Delta Level Shift are the prerequisites for this anomaly, but the value of Delta Level Shift cannot be inferred in this example unless both sensors register a level shift.

[33]"Spike seen in sensor 327 or 328 only, with no change in steady state pressures or pressure difference indicates no real rotor motion and possible anomaly in omni seal or sensor itself." [7]

A "default as superset" assumption that "Delta Level Shift" has value "Zero" breaks the logjam, but also runs afoul the expert's determination that "Delta Level Shift should not be zero" [7] except in certain cases. Using the Bayesian Forest's "default as negation" approach, the inference engine assumes the necessary value to allow its reasoning to continue.

5. CONCLUSIONS

This research develops a viable knowledge acquisition and maintenance tool and its associated methodology which together implement the new Bayesian Forest knowledge model. This new tool, MACK, guarantees the consistency of the data stored in a Bayesian Forest's knowledge base as it is both acquired and later maintained. Moreover, this tool has been applied to a real-world domain—NASA's Post-Test Diagnostic System—which supports Space Shuttle main engine analysis.

MACK, which is implemented on an explicit object-oriented analytical foundation, contains routines designed automatically and incrementally to confirm the consistency of the knowledge being received from the expert and provide him/her with natural assistance in the transfer of knowledge. Regular incremental checks preserve both probabilistic validity and logical consistency by flagging the inconsistent data points to the expert as they are entered and presumably under his/her current consideration. Such checking guards against expert oversight—*e.g.*, the "Whoops! I forgot to run the consistency checker again!" phenomenon—and helps prevent information overload since there can be at most five adjustments required of the expert immediately after any given run of the consistency checking module.[34]

[34]The consistency checking routine runs after each addition, removal or edit of *any* support node, following the removal an instantiation node, and after receipt of the fifth consecutive instantiation.

The tool is able to accept and manipulate time-dependent data which is both common and required not only in the PTDS domain modelled herein, but in many other real-world applications as well. Moreover, this capability will prove crucial to any eventual efforts to operate a Bayesian Forest inferencing mechanism in real-time or near real-time. In addition, we have determined the Bayesian Forest's available methods for dealing with the incomplete information which grant it its flexibility as a knowledge representation. "Default as negation" is the preferred mechanism as it preserves all of the rules and data interactions expressly catalogued by the expert and the ability of the forthcoming Explanation & Interpretation Facility to explain the system's results. This work breaks ground by integrating aspects of three disparate reasoning schemes—probabilistic reasoning, temporal reasoning, reasoning with defaults—into the Bayesian Forest model, particularly as they touch upon knowledge acquisition.

In order to implement the MACK tool properly to guarantee consistency of the knowledge, we had to formalize the notion of consistency for Bayesian Forests and then determine the necessary conditions and constraints. The constraints ensure the proper relationships between the forest's instantiation and support nodes are in force at all times. This includes algorithms both to detect constraint violations and to facilitate corrections.[35]

[35]An interesting challenge in the enumeration of these constraints lay in the development of the mathematical specifications from which they were built. For example, Constraint 5's cycles can be of any length ≥ 2 and there are a combinatorial number of support node permutations for these cycles based on Stirling numbers of the First Kind [68], [32]. Meanwhile, the groups of compatible support nodes which must all sum to less than 1 in Constraint 8 are also of indeterminate size. Automating this constraint by itself required separate routines to find and define five distinct sets of three of the different object types that make up a forest in for each sum! After that, normalization was little more than an afterthought.

We have availed ourselves of the Bayesian Forest's computational efficiencies without sacrificing the increased structural flexibility they afford both for reasoning with uncertainty and when compared to other Bayesian inferencing methods. We evidence this by the construction of a Bayesian Forest for the Post-Test Diagnostics System. Continued work in the application domain by PESKI and Bayesian Forest researchers promises to facilitate on-going PTDS knowledge acquisition within NASA as well as to provide the agency with novel, probabilistic reasoning alternatives.

MACK completes PESKI's Knowledge Acquisition and Maintenance module (see Figure 1.2). With that foundation established by this work, follow-on research will now concentrate on the development of the other components of the PESKI architecture, namely the Inference Engine and, just as importantly, the Explanation & Interpretation Facility. Future directions for work with the MACK tool include its application to a new domain, one for which there has been no previous knowledge engineering, and development of a graphical user interface better to interact with the expert. This could include, but is not limited to, "point-and-click" Bayesian Forest construction and on-screen representations of the forest's inconsistencies which should facilitate the expert's comprehension of the knowledge model.

Areas of particular interest for the inference engine will be its abilities to reason with defaults, to unify time and uncertainty for diagnosis [55], and to function over large data sets using techniques such as genetic algorithms [56]. Each of these areas promises only to augment Bayesian Forests which are already a powerful knowledge representation. The feedback module's efforts should focus on its ability to explain to the expert the

assumptions made during inferencing and efficiently to incorporate the expert's revisions into the knowledge base via MACK or other means. Ultimately, this module must be able not only to explain results at the expert's level, but also to interpret for the end-user whose expertise will be minimal.

Together with MACK, these components and the eventual Natural Language Interface, by allowing the expert to enter his/her data, conduct verification test runs, and receive from those tests an explanation detailed enough to allow the expert to refine and adjust the knowledge base appropriately, will establish Bayesian Forests as a front-line computing methodology for reasoning under uncertainty.

A. BAYESIAN FOREST MATHEMATICAL SPECIFICATION[36]

Component

name: seq *char*

Value

name: seq *char*

All Instantiations

$all\_instantiations: Component \leftrightarrow Value$

INODE

All_Instantiations
component: Component
value: Value

$(component, value) \in all\_instantiations$

Spt List

$entries: \mathcal{P}\ (INODE)$

All Supports

$all\_supports: Spt\_List \rightarrow INODE$

[36]See [26], [66], [46] for a more complete description of the *Z* notation used in this appendix.

SPTNODE

$$
\begin{array}{l}
All\_Supports \\
head:\ INODE \\
tail:\ Spt\_List \\
probability:\ \mathbb{R} \\
probability\_range:\ \mathbb{N} \\
\hline
(tail, head) \in all\_supports \\
probability \geq 0.0 \\
probability \leq 1.0 \\
probability\_range \leq 4
\end{array}
$$

Bayesian Forest

$$
\begin{array}{l}
instantiations:\ \mathbb{P}\ (INODE) \\
supports:\ \mathbb{P}\ (SPTNODE) \\
\hline
\forall\ i : instantiations, \exists\ s : supports \bullet i = s.head \\
\forall\ s : supports \bullet (s.tail.entries \cup \{s.head\}) \subseteq instantiations \\
\forall\ i,j :\ instantiations \bullet ((i \neq j) \wedge (i.component = j.component)) \Rightarrow (i.value \neq j.value) \\
\forall\ s,t : supports \bullet \quad (s.head = t.head) \\
\qquad\qquad \Rightarrow \\
\qquad (\exists\ i,j: instantiations \bullet ((i.component = j.component) \wedge (i \neq j) \\
\qquad\qquad \wedge (i \in s.tail.entries) \wedge (j \in t.tail.entries))) \\
\forall\ \Lambda : \mathbb{P}\ (SPTNODE), \nexists\ \mathrm{Y} : \mathrm{seq}\ \Lambda \bullet \\
\quad ((\#\ \mathrm{Y} = \#\ \Lambda) \\
\quad \wedge \\
\quad (\forall\ s : \Lambda, \exists\ (n,s) : \mathrm{Y}) \\
\quad \wedge \\
\quad (\exists\ (n,p), (m,q) : \mathrm{Y} \bullet ((m > n) \wedge (p.head \in q.tail.entries)))) \\
\forall\ i,j : instantiations, s : supports \bullet \\
\quad (\{i,j\} \subseteq (s.tail.entries \cup \{s.head\})) \Rightarrow (i.component \neq j.component) \\
\forall\ s : supports \bullet (\forall\ z : s.tail.entries \bullet z.component \neq s.head.component) \\
\forall\ c: Component, k: instantiations \bullet \\
\quad \nexists\ \Theta : \mathbb{P}\ (SPTNODE); \Omega,\Psi,\Phi : \mathbb{P}\ (INODE); \Gamma : \mathbb{P}\ (Component) \bullet \\
\qquad (\Theta = \{x: supports \mid ((c = x.head.component) \wedge ((k \in x.tail.entries) \\
\qquad\qquad \vee (\forall\ l: x.tail.entries, k.component \neq l.component)))\}) \\
\qquad \wedge \\
\qquad (\Phi = (\bigcup_{\forall y \in \Theta} y.tail.entries) \setminus \{k\}) \\
\qquad \wedge \\
\qquad (\Gamma = \{b: Component \mid \#(\{b\} \lhd \Phi) \geq 2\}) \\
\qquad \wedge \\
\qquad (\Psi = \{i: \Phi \mid i.component \in \Gamma\}) \\
\qquad \wedge \\
\qquad ((\#\ \Omega = \#\ \Gamma) \wedge (\Omega \subseteq \Psi) \wedge (\forall\ x: \Gamma, \exists\ y: \Omega \bullet (x = y.component))) \\
\qquad \wedge \\
\qquad (\sum_{\forall q \in \{z: \Theta \mid \Omega \cap z.tail.entries = \emptyset\}} q.probability > 1.0)
\end{array}
$$

B. SELECTED EXCERPTS OF KNOWLEDGE ACQUISITION INTERVIEWS

This appendix juxtaposes the output text from Aerojet Propulsion Division's HPOTP expert system and the Bayesian Forest support node(s) developed in this work with the actual input from the expert [7]. For each of the five anomalies highlighted herein, we identify first the expertise, followed by the Aerojet anomaly(-ies) and Bayesian Forest support conditions in that order.

========================

Anomaly 5.06.1:

- Expert Input from knowledge acquisition interview transcripts:
  "Seeing change in one, [but] not the other probably is due to static seal in housing or pressure shift not associated with real rotor motion. It probably is not a sensor problem."

- Anomaly Report Text:
  "Spike seen in sensor <327|328> only, with no change in steady state pressures or pressure difference. Possible sensor or omni seal anomaly. No real rotor motion."

- Bayesian Forest Support Conditions:
  - It is probable that the Anomaly 5.06.1 is Found depending upon . . .
    PID 327 = Spike -
    PID 328 = Spike 0
    Delta Level Shift for PIDs 327 & 328 = Zero

  - It is probable that the Anomaly 5.06.1 is Found depending upon . . .
    PID 327 = Spike +
    PID 328 = Spike 0
    Delta Level Shift for PIDs 327 & 328 = Zero

- It is probable that the Anomaly 5.06.1 is Found depending upon . . .
  - PID 328 = Spike -
  - PID 327 = Spike 0
  - Delta Level Shift for PIDs 327 & 328 = Zero

- It is probable that the Anomaly 5.06.1 is Found depending upon . . .
  - PID 328 = Spike +
  - PID 327 = Spike 0
  - Delta Level Shift for PIDs 327 & 328 = Zero

Anomaly 5.06.2:

o Expert Input from knowledge acquisition interview transcripts:

"Level change in one and not in the other . . . . I don't believe we can see cup seal/washer failures or problems. Changes were made to the design to eliminate this type of problem. Also it is highly unlikely that a piece (of seal) could migrate to a pressure opening an effect that pressure. I would be skeptical that it is a cup washer, more likely it is an omni seal or a sensor problem."

o Anomaly Report Text:

"Level shift seen in sensor <327|328> only. Possible sensor problem, omni seal leakage problem. No real rotor motion."

o Bayesian Forest Support Conditions:

- It is possible that the Anomaly 5.06.2 is Found depending upon . . .
  - PID 327 = Level Shift -
  - PID 328 = Level Shift 0

- It is possible that the Anomaly 5.06.2 is Found depending upon . . .
  - PID 327 = Level Shift +
  - PID 328 = Level Shift 0

- It is possible that the Anomaly 5.06.2 is Found depending upon . . .
  - PID 328 = Level Shift -
  - PID 327 = Level Shift 0

- It is possible that the Anomaly 5.06.2 is Found depending upon . . .
  - PID 328 = Level Shift +
  - PID 327 = Level Shift 0

Anomaly 5.06.4:

- Expert Input from knowledge acquisition interview transcripts:
  "See the delta is 327-328 so if this one goes up and this one goes down than this should go up and its possibly anomalous rotor motion considering this is constant power level."

- Anomaly Report Text:
  "Possible HPOTP anomalous rotor motion."

- Bayesian Forest Support Conditions:
  - It is possible that the Anomaly 5.06.4 is Found depending upon . . .
    PID 327 = Level Shift -
    PID 328 = Level Shift +
    Delta Level Shift for PIDs 327 & 328 = Negative

  - It is possible that the Anomaly 5.06.4 is Found depending upon . . .
    PID 327 = Level Shift +
    PID 328 = Level Shift -
    Delta Level Shift for PIDs 327 & 328 = Positive

Anomalies 5.15.1 & 5.15.2:

- Expert Input from knowledge acquisition interview transcripts:

  "951 is one of the [sensors] that addresses the pressure in the LOX [liquid oxygen] drain. 1187 is the temperature that I showed you that you can pick whether it's a new pump or not. 95% of the time it runs at 160 [psia] upwards to 450. I think from an erratic criteria needs to be able to discriminate. The thing that bothers me here is that 951 could be erratic for cause and it would not necessarily cause 1187 temperature measurement to be erratic. So should not try to couple the two measurements. The thing to do would be to classify either or both as being erratic and go from there. I believe that I would do 951 like we did the other pressures. Establish the data base and you compare each test against that database. You also analyze for erratic behavior and if it by itself falls out than flag it. The temperature of the erratic test is appropriate. There are two characteristics and one or the other is always there. What you look for is something different."

- Anomaly Report Text:

  "HPOTP erratic primary pump seal drain pressure may indicate sensor problem or seal anomaly. No effect seen in drain temperature."

- Bayesian Forest Support Conditions:
  - It is possible that the Anomaly 5.15.1 is Found depending upon . . .
    PID 951/952/953 = Erratic
    PID 1187 = Nominal

  - It is possible that the Anomaly 5.15.1 is Found depending upon . . .
    PID 951/952/953 = Spike
    PID 1187 = Nominal

  - It is possible that the Anomaly 5.15.2 is Found depending upon . . .
    PID 1187 = Erratic
    PID 951/952/953 = Nominal

  - It is possible that the Anomaly 5.15.2 is Found depending upon . . .
    PID 1187 = Spike
    PID 951/952/953 = Nominal

C. MACK SESSION TRANSCRIPT

Welcome to M.A.C.K. -- the Bayesian Forest Module for the Acquisition of Consistent Knowledge!!

0 - Generate new Bayesian Forest
1 - Edit existing Bayesian Forest
2 - Display current Bayesian Forest
3 - Load Bayesian Forest from file
4 - Save Bayesian Forest to file
5 - Check Forest Consistency
6 - Run Bayesian Forest Belief Revision Program
7 - Delete the current Bayesian Forest
8 - Exit Bayesian Forest program

Choice: 0

Growing trees for your Bayesian Forest.

Please select from the following menu:
Initializing forest.

Instantiations:
0 - Add new instantiation
1 - Delete instantiation

Support Conditions:
2 - Add new support condition
3 - Edit existing support condition
4 - Delete support condition

5 - Return to main menu

Choice: 0

Please pick a component to instantiate:

Empty NameTable.
0 -- Add new component
-1 -- Abort
Choice: 0

Please enter the name of the new component:
Item Name: PID 990 Peak
PID 990 Peak is a new component.

New INODE created: PID 990 Peak = No value given!!

REMINDER:

Bayesian Forest Variables are persistent and mutually exclusive.

In other words, they take on 1, and only 1, of their possible values.

Obviously, this will not accommodate variables that change over time.

Can the value of PID 990 Peak vary with time? Y / N _y_

How many times might PID 990 Peak change? _5_

Would you like to add another component?

_y_

Please enter the name of the new component:

Item Name: _PID 990 Equilibrium_

PID 990 Equilibrium is a new component.

New INODE created: PID 990 Equilibrium = No value given!!

REMINDER:

Bayesian Forest Variables are persistent and mutually exclusive.

In other words, they take on 1, and only 1, of their possible values.

Obviously, this will not accommodate variables that change over time.

Can the value of PID 990 Equilibrium vary with time? Y / N _y_

How many times might PID 990 Equilibrium change? _3_

Would you like to add another component?

_y_

Please enter the name of the new component:

Item Name: _Anomaly 990 Shift_

Anomaly 990 Shift is a new component.

New INODE created: Anomaly 990 Shift = No value given!!

REMINDER:

Bayesian Forest Variables are persistent and mutually exclusive.

In other words, they take on 1, and only 1, of their possible values.

Obviously, this will not accommodate variables that change over time.

Can the value of Anomaly 990 Shift vary with time? Y / N _y_

How many times might Anomaly 990 Shift change? _5_

Would you like to add another component?

_n_

Please pick a component to instantiate:

1 -- Anomaly 990 Shift

2 -- PID 990 Equilibrium
3 -- PID 990 Peak
0 -- Add new component
-1 -- Abort
Choice: 3

Creating new instantiation for PID 990 Peak
1 -- PID 990 Peak = No value given!!

Do you wish to enter a new legal value for PID 990 Peak [Enter 0]
or choose a pre-existing value [Enter 1]?
0
Enter the new value for PID 990 Peak:
Value: Nominal
New INODE created: PID 990 Peak = Nominal
INODE PID 990 Peak = No value given!! deleted

Instantiations:
0 - Add new instantiation
1 - Delete instantiation

Support Conditions:
2 - Add new support condition
3 - Edit existing support condition
4 - Delete support condition

5 - Return to main menu

Choice: 2

Please pick a component:

1 -- Anomaly 990 Shift
2 -- PID 990 Equilibrium
3 -- PID 990 Peak
0 -- Abort
Choice: 3

Please pick an instantiation of PID 990 Peak:
1 -- PID 990 Peak = Nominal
0 -- Abort
Choice: 1

At present, the PID 990 Peak's being Nominal depends upon the following:

At present, PID 990 Peak's being Nominal depends upon the following sets of conditions:

No support conditions!

Enter 0 to add new support conditions for
PID 990 Peak's being Nominal

Otherwise, enter 1 to quit
_0_

PID 990 Peak's being Nominal
can depend upon which of the following components:
1 -- Anomaly 990 Shift
2 -- PID 990 Equilibrium
3 -- PID 990 Peak
0 -- None of the Above Components
Choice: _0_

Please complete the sentence below from the following list of choices:

0 -- inconceivable
1 -- not likely
2 -- possible
3 -- probable
4 -- almost certain

It is ____________ that the PID 990 Peak is Nominal depending upon . . .
Nothing!

Choice: _2_

Presently, this condition

It is virtually impossible that the Anomaly 990 Shift is No value given!! depending upon . . .
Nothing!

is _less_than_ inconceivable; it's virtually impossible!!
If this is inaccurate, you may wish to edit this support condition.

Presently, this condition

It is virtually impossible that the PID 990 Equilibrium is No value given!! depending upon . . .
Nothing!

is _less_than_ inconceivable; it's virtually impossible!!
If this is inaccurate, you may wish to edit this support condition.

Instantiations:
0 - Add new instantiation
1 - Delete instantiation

Support Conditions:
2 - Add new support condition
3 - Edit existing support condition
4 - Delete support condition

5 - Return to main menu

Choice: _0_

Please pick a component to instantiate:

1 -- Anomaly 990 Shift
2 -- PID 990 Equilibrium
3 -- PID 990 Peak
0 -- Add new component
-1 -- Abort

Choice: _3_

Creating new instantiation for PID 990 Peak
1 -- PID 990 Peak = Nominal

Do you wish to enter a new legal value for PID 990 Peak [Enter 0]
or choose a pre-existing value [Enter 1]?
_0_

Enter the new value for PID 990 Peak:
Value: _Out of Family -- High_
New INODE created: PID 990 Peak = Out of Family -- High

Instantiations:
0 - Add new instantiation
1 - Delete instantiation

Support Conditions:
2 - Add new support condition
3 - Edit existing support condition
4 - Delete support condition

5 - Return to main menu

Choice: _0_

Please pick a component to instantiate:

1 -- Anomaly 990 Shift
2 -- PID 990 Equilibrium
3 -- PID 990 Peak

0 -- Add new component
-1 -- Abort
Choice: _3_

Creating new instantiation for PID 990 Peak
1 -- PID 990 Peak = Nominal
2 -- PID 990 Peak = Out of Family -- High

Do you wish to enter a new legal value for PID 990 Peak [Enter 0]
or choose a pre-existing value [Enter 1]?
_0_

Enter the new value for PID 990 Peak:
Value: _Out of Family -- Low_
New INODE created: PID 990 Peak = Out of Family -- Low

Instantiations:
0 - Add new instantiation
1 - Delete instantiation

Support Conditions:
2 - Add new support condition
3 - Edit existing support condition
4 - Delete support condition

5 - Return to main menu

Choice: _0_

Please pick a component to instantiate:

1 -- Anomaly 990 Shift
2 -- PID 990 Equilibrium
3 -- PID 990 Peak
0 -- Add new component
-1 -- Abort
Choice: _2_

Creating new instantiation for PID 990 Equilibrium
1 -- PID 990 Equilibrium = No value given!!

Do you wish to enter a new legal value for PID 990 Equilibrium [Enter 0]
or choose a pre-existing value [Enter 1]?
_1_

Please instantiate PID 990 Equilibrium from this menu:

1 -- Nominal
2 -- Out of Family -- High
3 -- Out of Family -- Low
0 -- Abort

Choice: 1

New INODE created: PID 990 Equilibrium = Nominal
INODE PID 990 Equilibrium = No value given!! deleted

Instantiations:
0 - Add new instantiation
1 - Delete instantiation

Support Conditions:
2 - Add new support condition
3 - Edit existing support condition
4 - Delete support condition

5 - Return to main menu

Choice: 0

Please pick a component to instantiate:

1 -- Anomaly 990 Shift
2 -- PID 990 Equilibrium
3 -- PID 990 Peak
0 -- Add new component
-1 -- Abort

Choice: 2

Creating new instantiation for PID 990 Equilibrium
1 -- PID 990 Equilibrium = Nominal

Do you wish to enter a new legal value for PID 990 Equilibrium [Enter 0]
or choose a pre-existing value [Enter 1]?
0

Enter the new value for PID 990 Equilibrium:
Value: Out of Family
New INODE created: PID 990 Equilibrium = Out of Family

Instantiations:
0 - Add new instantiation
1 - Delete instantiation

Support Conditions:
2 - Add new support condition
3 - Edit existing support condition
4 - Delete support condition

5 - Return to main menu

Choice: 0

Please pick a component to instantiate:

1 -- Anomaly 990 Shift
2 -- PID 990 Equilibrium
3 -- PID 990 Peak
0 -- Add new component
-1 -- Abort

Choice: _1_

Creating new instantiation for Anomaly 990 Shift
1 -- Anomaly 990 Shift = No value given!!

Do you wish to enter a new legal value for Anomaly 990 Shift [Enter 0]
or choose a pre-existing value [Enter 1]?
_0_

Enter the new value for Anomaly 990 Shift:
Value: _Found_
New INODE created: Anomaly 990 Shift = Found
INODE Anomaly 990 Shift = No value given!! deleted

This Bayesian Forest is currently inconsistent.
Is it correct that
Anomaly 990 Shift being Found
does not depend on anything else? Y/N _n_

Please edit the conditions for
Anomaly 990 Shift being Found accordingly.
Bayesian Forest fails consistency checking.

Instantiations:
0 - Add new instantiation
1 - Delete instantiation

Support Conditions:
2 - Add new support condition
3 - Edit existing support condition
4 - Delete support condition

5 - Return to main menu

Choice: _0_

Please pick a component to instantiate:

1 -- Anomaly 990 Shift
2 -- PID 990 Equilibrium
3 -- PID 990 Peak
0 -- Add new component
-1 -- Abort

Choice: _1_

Creating new instantiation for Anomaly 990 Shift
1 -- Anomaly 990 Shift = Found

Do you wish to enter a new legal value for Anomaly 990 Shift [Enter 0]
or choose a pre-existing value [Enter 1]?
0

Enter the new value for Anomaly 990 Shift:
Value: Not Found
New INODE created: Anomaly 990 Shift = Not Found

Instantiations:
0 - Add new instantiation
1 - Delete instantiation

Support Conditions:
2 - Add new support condition
3 - Edit existing support condition
4 - Delete support condition

5 - Return to main menu

Choice: 2

Please pick a component:

1 -- Anomaly 990 Shift
2 -- PID 990 Equilibrium
3 -- PID 990 Peak
0 -- Abort
Choice: 1

Please pick an instantiation of Anomaly 990 Shift:
1 -- Anomaly 990 Shift = Found
2 -- Anomaly 990 Shift = Not Found
0 -- Abort
Choice: 1

At present, the Anomaly 990 Shift's being Found depends upon the following:

At present, Anomaly 990 Shift's being Found depends upon the following sets of conditions:

No support conditions!

Enter 0 to add new support conditions for
Anomaly 990 Shift's being Found
Otherwise, enter 1 to quit
0

Anomaly 990 Shift's being Found
can depend upon which of the following components:

1 -- Anomaly 990 Shift
2 -- PID 990 Equilibrium
3 -- PID 990 Peak
0 -- None of the Above Components

Choice: 2

1 -- Nominal
2 -- Out of Family
0 -- None of the Above; Abort

Choice: 2

It is < UNDEFINED > that the Anomaly 990 Shift @ T_i is Found depending upon . . .
PID 990 Equilibrium @ T_i = Out of Family

New addition, PID 990 Equilibrium, is time-dependent.
We should read its value, Out of Family, from which time interval?
-1 -- Time period immediately preceding Anomaly 990 Shift = Found
0 -- The same time period as Anomaly 990 Shift = Found
1 -- Time period immediately after Anomaly 990 Shift = Found

0

Presently, this condition holds that Anomaly 990 Shift's being Found
can depend upon the following:

PID 990 Equilibrium = Out of Family

Do you wish to extend this condition? Y / N y

Anomaly 990 Shift's being Found
can depend upon which of the following components:

3 -- PID 990 Peak

Choice: 3

1 -- Nominal
2 -- Out of Family -- High
3 -- Out of Family -- Low
0 -- None of the Above; Abort

Choice: 2

It is < UNDEFINED > that the Anomaly 990 Shift @ T_i is Found depending upon . . .
PID 990 Equilibrium @ T_i = Out of Family
PID 990 Peak @ T_i = Out of Family -- High

New addition, PID 990 Peak, is time-dependent.
We should read its value, Out of Family -- High, from which time interval?
-1 -- Time period immediately preceding Anomaly 990 Shift = Found
0 -- The same time period as Anomaly 990 Shift = Found
1 -- Time period immediately after Anomaly 990 Shift = Found

0

Presently, this condition holds that Anomaly 990 Shift's being Found can depend upon the following:

PID 990 Equilibrium = Out of Family
PID 990 Peak = Out of Family -- High

Do you wish to extend this condition? Y / N _n_

Please complete the sentence below from the following list of choices:

0 -- inconceivable
1 -- not likely
2 -- possible
3 -- probable
4 -- almost certain

It is ____________ that the Anomaly 990 Shift is Found depending upon . . .
PID 990 Equilibrium = Out of Family
PID 990 Peak = Out of Family -- High
Choice: _3_

This Bayesian Forest is currently inconsistent.
Is it correct that
Anomaly 990 Shift being Not Found
does not depend on anything else? Y/N _y_

Please complete the sentence below from the following list of choices:

0 -- inconceivable
1 -- not likely
2 -- possible
3 -- probable
4 -- almost certain

It is ____________ that the Anomaly 990 Shift is Not Found depending upon . . .
Nothing!
Choice: _3_

This Bayesian Forest is currently inconsistent.
Is it correct that
PID 990 Equilibrium being Nominal
does not depend on anything else? Y/N _y_

Please complete the sentence below from the following list of choices:

0 -- inconceivable
1 -- not likely
2 -- possible
3 -- probable
4 -- almost certain

It is ____________ that the PID 990 Equilibrium is Nominal depending upon . . .

Nothing!

Choice: 3

This Bayesian Forest is currently inconsistent.
Is it correct that

PID 990 Equilibrium being Out of Family

does not depend on anything else? Y/N y

Please complete the sentence below from the following list of choices:

0 -- inconceivable
1 -- not likely
2 -- possible
3 -- probable
4 -- almost certain

It is ____________ that the PID 990 Equilibrium is Out of Family depending upon . . .

Nothing!

Choice: 1

This Bayesian Forest is currently inconsistent.
Is it correct that

PID 990 Peak being Out of Family -- High

does not depend on anything else? Y/N y

Please complete the sentence below from the following list of choices:

0 -- inconceivable
1 -- not likely
2 -- possible
3 -- probable
4 -- almost certain

It is ____________ that the PID 990 Peak is Out of Family -- High depending upon . . .

Nothing!

Choice: 1

This Bayesian Forest is currently inconsistent.
Is it correct that

PID 990 Peak being Out of Family -- Low

does not depend on anything else? Y/N y

Please complete the sentence below from the following list of choices:

0 -- inconceivable
1 -- not likely
2 -- possible
3 -- probable
4 -- almost certain

It is ____________ that the PID 990 Peak is Out of Family -- Low depending upon . . .
Nothing!
Choice: _1_

This Bayesian Forest is inconsistent.
Currently, support ranges overlap. Adjusting ranges for consistency . . .
Conditions were:

It is probable that the Anomaly 990 Shift @ T_i is Found depending upon . . .
PID 990 Equilibrium @ T_i = Out of Family
PID 990 Peak @ T_i = Out of Family -- High

It is probable that the Anomaly 990 Shift is Not Found depending upon . . .
Nothing!

New conditions are:

It is possible that the Anomaly 990 Shift @ T_i is Found depending upon . . .
PID 990 Equilibrium @ T_i = Out of Family
PID 990 Peak @ T_i = Out of Family -- High

It is possible that the Anomaly 990 Shift is Not Found depending upon . . .
Nothing!

This Bayesian Forest is inconsistent.
Currently, support ranges overlap. Adjusting ranges for consistency . . .
Conditions were:

It is probable that the PID 990 Equilibrium is Nominal depending upon . . .
Nothing!

It is not likely that the PID 990 Equilibrium is Out of Family depending upon . . .
Nothing!

New conditions are:

It is probable that the PID 990 Equilibrium is Nominal depending upon . . .
Nothing!

It is not likely that the PID 990 Equilibrium is Out of Family depending upon . . .
Nothing!

This Bayesian Forest is inconsistent.
Currently, support ranges overlap. Adjusting ranges for consistency . . .
Conditions were:

It is possible that the PID 990 Peak is Nominal depending upon . . .
Nothing!

It is not likely that the PID 990 Peak is Out of Family -- High depending upon . . .
Nothing!

It is not likely that the PID 990 Peak is Out of Family – Low depending upon . . .
Nothing!

New conditions are:

It is possible that the PID 990 Peak is Nominal depending upon . . .
Nothing!

It is not likely that the PID 990 Peak is Out of Family – High depending upon . . .
Nothing!

It is not likely that the PID 990 Peak is Out of Family – Low depending upon . . .
Nothing!

Instantiations:
0 - Add new instantiation
1 - Delete instantiation

Support Conditions:
2 - Add new support condition
3 - Edit existing support condition
4 - Delete support condition

5 - Return to main menu

Choice: 2

Please pick a component:

1 -- Anomaly 990 Shift
2 -- PID 990 Equilibrium
3 -- PID 990 Peak
0 -- Abort

Choice: 1

Please pick an instantiation of Anomaly 990 Shift:
1 -- Anomaly 990 Shift = Found
2 -- Anomaly 990 Shift = Not Found
0 -- Abort

Choice: 1

At present, the Anomaly 990 Shift's being Found depends upon the following:

At present, Anomaly 990 Shift's being Found depends upon the following sets of conditions:

Support Node #1:
PID 990 Equilibrium = Out of Family
PID 990 Peak = Out of Family – High

Enter 0 to add new support conditions for
Anomaly 990 Shift's being Found

Otherwise, enter 1 to quit
0

Anomaly 990 Shift's being Found
can depend upon which of the following components:
1 -- Anomaly 990 Shift
2 -- PID 990 Equilibrium
3 -- PID 990 Peak
0 -- None of the Above Components
Choice: 3

1 -- Nominal
2 -- Out of Family -- High
3 -- Out of Family -- Low
0 -- None of the Above; Abort
Choice: 3

It is < UNDEFINED > that the Anomaly 990 Shift @ T_i is Found depending upon . . .
PID 990 Peak @ T_i = Out of Family -- Low

New addition, PID 990 Peak, is time-dependent.
We should read its value, Out of Family -- Low, from which time interval?
-1 -- Time period immediately preceding Anomaly 990 Shift = Found
0 -- The same time period as Anomaly 990 Shift = Found
1 -- Time period immediately after Anomaly 990 Shift = Found
0

Presently, this condition holds that Anomaly 990 Shift's being Found
can depend upon the following:

PID 990 Peak = Out of Family -- Low

Do you wish to extend this condition? Y / N y

Anomaly 990 Shift's being Found
can depend upon which of the following components:

2 -- PID 990 Equilibrium
Choice: 2

1 -- Nominal
2 -- Out of Family
0 -- None of the Above; Abort
Choice: 1

It is < UNDEFINED > that the Anomaly 990 Shift @ T_i is Found depending upon . . .
PID 990 Peak @ T_i = Out of Family -- Low
PID 990 Equilibrium @ T_i = Nominal

New addition, PID 990 Equilibrium, is time-dependent.
We should read its value, Nominal, from which time interval?

-1 -- Time period immediately preceding Anomaly 990 Shift = Found
0 -- The same time period as Anomaly 990 Shift = Found
1 -- Time period immediately after Anomaly 990 Shift = Found

0

Presently, this condition holds that Anomaly 990 Shift's being Found can depend upon the following:

PID 990 Peak = Out of Family -- Low
PID 990 Equilibrium = Nominal

Do you wish to extend this condition? Y / N n

Please complete the sentence below from the following list of choices:

0 -- inconceivable
1 -- not likely
2 -- possible
3 -- probable
4 -- almost certain

It is ____________ that the Anomaly 990 Shift is Found depending upon . . .
PID 990 Peak = Out of Family -- Low
PID 990 Equilibrium = Nominal
Choice: 3

Sum of the probabilities cannot exceed 1.0!
This Bayesian Forest is inconsistent.
Currently, support ranges overlap. Adjusting ranges for consistency . . .
Conditions were:

It is probable that the Anomaly 990 Shift @ T_i is Found depending upon . . .
PID 990 Peak @ T_i = Out of Family -- Low
PID 990 Equilibrium @ T_i = Nominal

It is possible that the Anomaly 990 Shift is Not Found depending upon . . .
Nothing!

New conditions are:

It is possible that the Anomaly 990 Shift @ T_i is Found depending upon . . .
PID 990 Peak @ T_i = Out of Family -- Low
PID 990 Equilibrium @ T_i = Nominal

It is possible that the Anomaly 990 Shift is Not Found depending upon . . .
Nothing!

Instantiations:
0 - Add new instantiation
1 - Delete instantiation

Support Conditions:

2 - Add new support condition
3 - Edit existing support condition
4 - Delete support condition

5 - Return to main menu

Choice: _2_

Please pick a component:

1 -- Anomaly 990 Shift
2 -- PID 990 Equilibrium
3 -- PID 990 Peak
0 -- Abort

Choice: _1_

Please pick an instantiation of Anomaly 990 Shift:

1 -- Anomaly 990 Shift = Found
2 -- Anomaly 990 Shift = Not Found
0 -- Abort

Choice: _1_

At present, the Anomaly 990 Shift's being Found depends upon the following:

At present, Anomaly 990 Shift's being Found depends upon the following sets of conditions:

Support Node #1:
PID 990 Equilibrium = Out of Family
PID 990 Peak = Out of Family -- High
Support Node #2:
PID 990 Peak = Out of Family -- Low
PID 990 Equilibrium = Nominal

Enter 0 to add new support conditions for
Anomaly 990 Shift's being Found
Otherwise, enter 1 to quit
_0_

Anomaly 990 Shift's being Found
can depend upon which of the following components:

1 -- Anomaly 990 Shift
2 -- PID 990 Equilibrium
3 -- PID 990 Peak
0 -- None of the Above Components

Choice: _2_

1 -- Nominal
2 -- Out of Family
0 -- None of the Above; Abort

Choice: _1_

It is < UNDEFINED > that the Anomaly 990 Shift @ T_i is Found depending upon . . .
PID 990 Equilibrium @ T_i = Nominal

New addition, PID 990 Equilibrium, is time-dependent.
We should read its value, Nominal, from which time interval?
-1 -- Time period immediately preceding Anomaly 990 Shift = Found
0 -- The same time period as Anomaly 990 Shift = Found
1 -- Time period immediately after Anomaly 990 Shift = Found
0

Presently, this condition holds that Anomaly 990 Shift's being Found can depend upon the following:

PID 990 Equilibrium = Nominal

Do you wish to extend this condition? Y / N n

Please complete the sentence below from the following list of choices:

0 -- inconceivable
1 -- not likely
2 -- possible
3 -- probable
4 -- almost certain

It is ____________ that the Anomaly 990 Shift is Found depending upon . . .
PID 990 Equilibrium = Nominal
Choice: 2

ERROR: Support conditions below are not mutually exclusive.

At present, Anomaly 990 Shift's being Found depends upon the following sets of conditions:

Support Node #1:
PID 990 Equilibrium = Out of Family
PID 990 Peak = Out of Family -- High
Support Node #2:
PID 990 Peak = Out of Family -- Low
PID 990 Equilibrium = Nominal
Support Node #3:
PID 990 Equilibrium = Nominal

This Bayesian Forest is currently inconsistent.
The following pair of conditions for Anomaly 990 Shift being Found are not mutually exclusive.

First Set:
PID 990 Peak = Out of Family -- Low
PID 990 Equilibrium = Nominal
Second Set:
PID 990 Equilibrium = Nominal
Does Anomaly 990 Shift's being Found

really depend upon _both_ sets of conditions? [Enter 0]
or
upon each set separately? [Enter 1]
Choice: _1_

Which of these conditions may we add to eliminate the overlap?
1 -- PID 990 Peak can be Nominal
2 -- PID 990 Peak can be Out of Family -- High
3 -- PID 990 Equilibrium can be Out of Family
4 -- PID 990 Equilibrium can be Out of Family
0 -- None of the Above
Choice: _2_

It is possible that the Anomaly 990 Shift @ T_i is Found depending upon . . .
PID 990 Equilibrium @ T_i = Nominal
PID 990 Peak @ T_i = Out of Family -- High

New addition, PID 990 Peak, is time-dependent.
We should read its value, Out of Family -- High, from which time interval?
-1 -- Time period immediately preceding Anomaly 990 Shift = Found
0 -- The same time period as Anomaly 990 Shift = Found
1 -- Time period immediately after Anomaly 990 Shift = Found
_0_

Sum of the probabilities cannot exceed 1.0!

This Bayesian Forest is inconsistent.
Currently, support ranges overlap. Adjusting ranges for consistency . . .
Conditions were:

It is possible that the Anomaly 990 Shift @ T_i is Found depending upon . . .
PID 990 Equilibrium @ T_i = Nominal
PID 990 Peak @ T_i = Out of Family -- High

It is possible that the Anomaly 990 Shift is Not Found depending upon . . .
Nothing!

New conditions are:

It is possible that the Anomaly 990 Shift @ T_i is Found depending upon . . .
PID 990 Equilibrium @ T_i = Nominal
PID 990 Peak @ T_i = Out of Family -- High

It is possible that the Anomaly 990 Shift is Not Found depending upon . . .
Nothing!

Instantiations:
0 - Add new instantiation
1 - Delete instantiation

Support Conditions:

2 - Add new support condition
3 - Edit existing support condition
4 - Delete support condition

5 - Return to main menu

Choice: _5_

0 - Generate new Bayesian Forest
1 - Edit existing Bayesian Forest
2 - Display current Bayesian Forest
3 - Load Bayesian Forest from file
4 - Save Bayesian Forest to file
5 - Check Forest Consistency
6 - Run Bayesian Forest Belief Revision Program
7 - Delete the current Bayesian Forest
8 - Exit Bayesian Forest program

Choice: _4_
Enter filename: _output.test_
Forest saved to file.

0 - Generate new Bayesian Forest
1 - Edit existing Bayesian Forest
2 - Display current Bayesian Forest
3 - Load Bayesian Forest from file
4 - Save Bayesian Forest to file
5 - Check Forest Consistency
6 - Run Bayesian Forest Belief Revision Program
7 - Delete the current Bayesian Forest
8 - Exit Bayesian Forest program

Choice: _8_

Cutting down the forest.
Goodbye.

D. TEMPORAL PARSING

This appendix outlines the methodology [7] by which a test run is parsed into the time slices discussed in Chapter 4. At present, MACK uses a similar method, but this capability is neither automated nor a part of the tool itself. Instead, we simply preprocess the temporal data by hand before entering it.

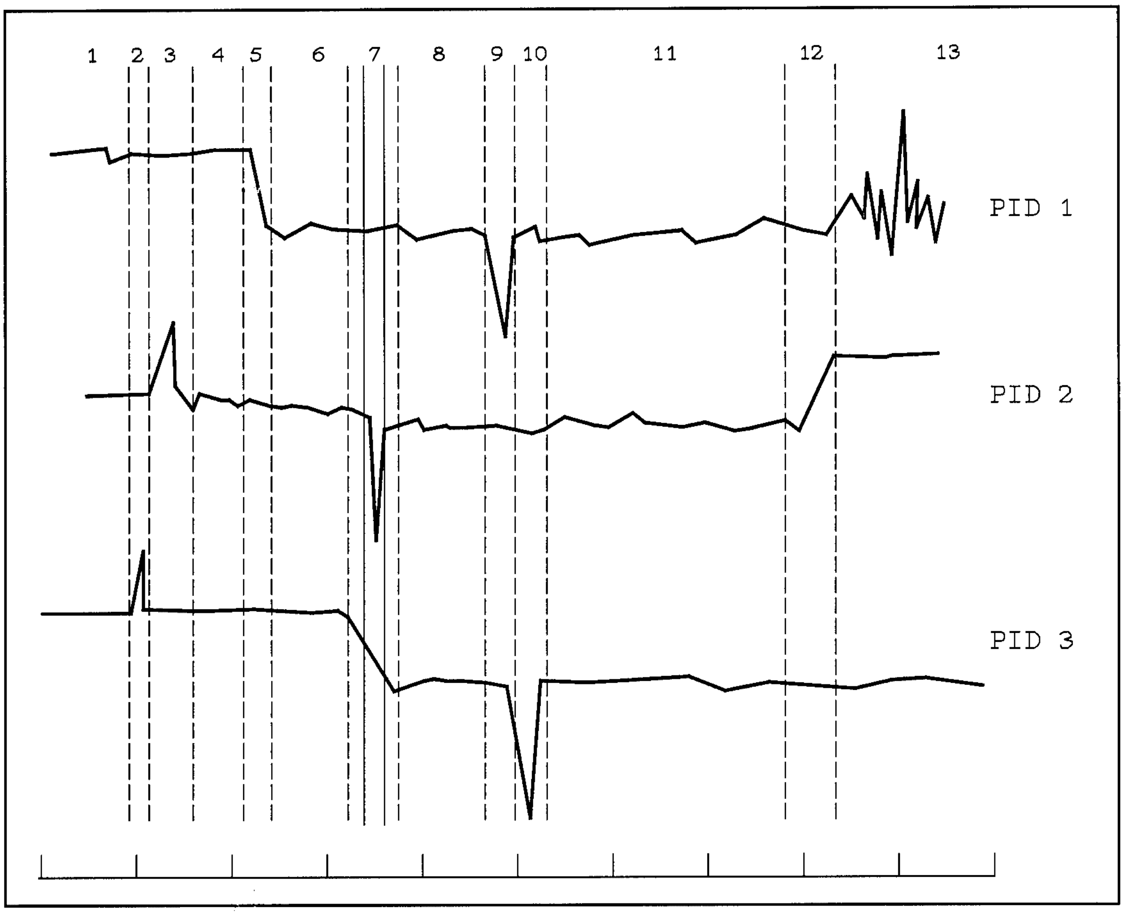

Figure D.1 Timeline Parsing Example

In Figure D.1 we have constructed a very simplistic example. The figure contai ns data streams from three hypothetical sensors over a ten-second period divided into one-second intervals. The algorithm assigns breakpoints in the timeline at the beginning and end of any event—*e.g.*, spikes, peaks, level shifts—from any of the sensors. The result is a series of intervals as shown.

Periods 1, 4, 6, 8 and 11 contain no significant events, *i.e.*, all sensors are nominal. Positive spikes/peaks occur in periods 2 and 3; negative spikes in 7, 9 and 10. We see level shifts during periods 5, 7 and 12, and erratic readings from PID 1 in the last interval. Notice that period 7, originally subdivided into three by the spike in PID 2, is unified to reflect the occurrence of a single level shift in PID 3 rather than three distinct shifts which would be recognizable by two small plateaux during the decline.

We consider events which start within one second of each other to be concurrent. Thus, the spikes in periods 2 and 3 are concurrent as are those in periods 9 and 10. PID 3's level shift in period 7 is, per force, concurrent with the spike in PID 2. Similarly, the shift during period 12 coincides with PID 1's erratic behavior.

E. GLOSSARY of TECHNICAL TERMS

Unless otherwise specified, the non-acronym terms and definitions contained in this glossary have been excerpted from either of the following sources: Rosenberg's *Dictionary of Artificial Intelligence and Robotics* [44] or Smith's *The Facts on File Dictionary of Artificial Intelligence* [63].

Abduction: An inference process that generates explanations. To a first approximation, it uses the following paradigm:

Given:	b
	a → b
Infer:	a

Unlike deduction, abduction is not a legal inference. However, despite the fact that abduction can lead to wrong conclusions, it is useful and often necessary.

Artificial Intelligence (AI): the capability of a device to perform functions that are normally associated with human intelligence such as reasoning, learning, and self-improvement.

Automated Reasoning: See Expert system.

Bayes' Rule: A theorem of probability that say if one knows, for example, how many patients with a given disease show a particular symptom, as well as the likelihood of the disease and of the symptom, then you can calculate the conditional probability of the disease given the symptom.

Bayesian Inference: Also known as statistical inference, one means by which knowledge-based systems can reason when uncertainty is involved. Given a hypothesis event H and an evidence event E, we obtain from an expert estimates of the prior probabilities P(H) and P(E), and the conditional probability P(E|H). From Bayes' Rule we obtain the probability of H given evidence E. In practical problems E may be any subset of all possible evidence events and H may be any subset of the set of all possible hypotheses. This tends to require a vast number of conditional probabilities to be calculated. An alternative approach is therefore to combine with a rule-based system where conditional probabilities are given only for each rule. This approach is adequate until one considers the local updating problem that arises when two or more inference rules make a conclusion about the same hypothesis.

Causal Analysis: used in credit (blame) assignment to track the probable causes of observed events.

Causal Model: a model where the causal relations among various actions and events are represented explicitly.

Certainty: the degree of confidence one has in a fact or relationship. As used in AI, contrasts with probability, which is the likelihood that an event will occur.

Certainty Factor: a numerical weight given to a fact or relationship indicating the confidence one has in the fact or relationship. These numbers behave differently that probability coefficients. In general, methods for manipulating certainty factors are more informal than approaches to combining probabilities. Most rule-based systems use certainty factors rather than probabilities. Synonymous with Confidence Factor.

Deduction: In formal logic the derivation of a logical consequence from a specific set of premises; a truth-preserving transformation of assertions.

Default: A value that is used when no other value is specified.

Default Reasoning [61]: Patterns of inference that permit the drawing of conclusions suggested but not entailed by their premises; Plays a particular role in two kinds of situations: those in which systems must reason from incomplete information and those in which systems must reason using uncertain rules.

Domain: a topical area or region of knowledge. Medicine, management, science, and engineering are very broad domains. Existing knowledge systems only provide competent advice within very narrowly defined domains.

Domain Expert: an individual who, through years of experience and training, has become extremely skilled at problem solving in a specific domain.

Domain Knowledge: knowledge about the problem domain, e.g., knowledge about geology in an expert system for discovering oil reserves.

Expert System: (1) a computer system that can perform at, or near, the level of a human expert. (2) any computer system developed by means of a loose collection of techniques associated with artificial intelligence research. Therefore, any computer system developed by means of an expert system building tool (even were the system to be so narrowly constrained that it could never be said to rival a human expert). (3) a computer program that performs a specialized, usually difficult professional task at the level of (or sometimes beyond the level of) a human expert. Because their functioning relies heavily on large bodies of knowledge, expert systems are sometimes known as knowledge-based systems. Since they are often used to assist the human expert, they are also known as intelligent assistants.

Expert System-Building Tool: the programming language and support package for building an expert systems.

Expertise: skill and knowledge possessed by humans resulting in performance far above the norm; consists of massive amounts of information combined with rules of thumb, simplifications, rare facts, and wise procedures in such a fashion that a person can analyze specific types of problems in an efficient way.

Explanation: information presented to justify a specific course of reasoning or action. In knowledge systems, a number of techniques that help a user understand what a system is doing. Many knowledge systems permit a user to ask "Why," "How," or "Explain." In each case, the system responds by revealing something about its assumptions or its inner reasoning.

Explanation Facility: the portion of an expert system that explains how solutions were arrived at and justifies the steps used in reaching them; keeps track of the reasoning paths used by the inference engine to reach its conclusions [53].

HPOTP: High-Pressure Oxidizer Turbopump

Inference: a process by which new facts are derived from known facts.

Inference Chain: the sequence of steps or rule applications utilized by a rule-based system to reach a conclusion.

Inference Engine: That portion of a knowledge system containing the inference and control strategies; includes various knowledge acquisition, explanation, and user interface subsystems. Inference engines are characterized by the inference and control strategies they use.

Instantiation: A pattern or formula in which the variables have been replaced by constants. The association of a particular individual with the characteristics of some class, or the assignment of particular values to the parameters of a procedure.

Integer Linear Programming [30]: The general problem of allocating limited resource among competing activities in the best possible, or optimal, integer-valued manner.

Intelligent system: See Expert system.

Knowledge: an integrated collection of facts and relationships which, when exercised, produces competent performance. The quantity and quality of knowledge possessed by an individual or a computer can be judged by the variety of situation in which the individual or program obtains successful results.

Knowledge Acquisition: the process of locating, collecting, and refining knowledge; may require interviews with experts, research in a library, or introspection. The individual doing this must convert the acquired knowledge into a form that can be used by a computer program.

Knowledge Base: Facts, assumptions, beliefs, heuristics, and expertise; methods of dealing with the data base to achieve desired results such as a diagnosis, interpretation, or solution to a problem.

Knowledge Engineer: An individual whose specialty is assessing problems, acquiring knowledge, and building expert systems.

Knowledge Engineering: The discipline of designing and building expert systems and other knowledge-based programs.

Knowledge Representation: A means for encoding and storing facts and relationships in a knowledge base. Semantic networks, object-attribute-value triplets, production rules, frames, and logical expressions are all ways to represent knowledge.

LOX: Liquid Oxygen

MACK: Module for the Acquisition of Consistent Knowledge. A knowledge acquisition tool designed to support Bayesian Forests.

Natural Language: a branch of artificial intelligence research that studies methods permitting computer systems to accept inputs and produce outputs in a conventional language like English.

NP [21]: The class of decision problems that can be solved in polynomial time by a non-deterministic computer. Most of the apparently intractable problems encountered in practice, when phrased as decision problems, belong to this class.

NP-Complete [21]: The equivalence class consisting of the hardest problems in NP.

NP-Hard [21]: Any decision problem, whether a member of NP or not, to which we can transform an NP-complete problem and, as a result, is at least as hard as the NP-complete problems.

Object-Oriented Techniques: Programming procedures based on the use of items called objects that communicate with one another via messages in the form of global broadcasts.

PESKI [53]: Probabilities, Expert Systems, Knowledge and Inference.

Probability: Various approaches to statistical inference used for determining the likelihood of a particular relationship. Expert systems have generally avoided probability and used confidence or certainty factors instead. See Certainty.

Probability Propagation: The adjusting of probabilities at the nodes in an inference net accounting for the effect of new information about the probability at a specific node.

Probabilistic Reasoning [29]: Mathematical theories based on probability and statistics are used to accept or reject proposed hypotheses and to draw other kinds of conclusions. These theories are quite well developed and computable in a straightforward numerical way. Thus, it is the design of the hypotheses and the use to which the conclusions are put that has more to do with AI than the actual method of reaching the conclusion.

PTDS: Post-Test Diagnostics System.

Real-Time: Pertaining to an application in which response to input is fast enough to affect subsequent input, such as a process control system or a computer-assisted instruction system.

Real-World Problem: A complex, practical problem having a solution that is useful in some cost-effective fashion.

Reasoner: See Inference Engine.

Simplex Method [72], [30]: A general algorithm for solving linear programming problems developed by George Dantzig in 1947.

SSME: Space Shuttle Main Engine

Stirling Numbers of the First Kind [68]: Related to binomial coefficients, these count the number of ways to arrange n objects into k cycles.

Stirling Numbers of the Second Kind [68]: Related to binomial coefficients, these count the number of ways to partition a set of n objects into k non-empty sets.

Tool: Computer software package that simplifies the effort involve din building an expert system; contains an inference engine and various user interface and knowledge acquisition aids, and lack of knowledge base.

Uncertainty: With expert systems, a value that cannot be determined during a consultation. Most expert systems can accommodate uncertainty by allowing the user to indicate if s/he does not know the answer.

BIBLIOGRAPHY

1. Allen, James F. "Maintaining knowledge about temporal intervals." *Communications of the ACM, 26 (11)*: 832-843, 1983.

2. Bacchus, Fahiem. "Representing and Reasoning with Probabilistic Knowledge." Research Report CS-88-31. Ph.D. dissertation. University of Alberta, Edmonton, Alberta, Fall 1988.

3. Bacchus, Fahiem. *Representing and Reasoning with Probabilistic Knowledge: A Logical Approach to Probabilities*. Cambridge, Massachusetts: The MIT Press, 1990.

4. Baenen, Eric Paul. *Generalized Probabilistic Reasoning and Empirical Studies on Computational Efficiency and Scalability*. M.S. thesis, AFIT/GCE/ENG/94D-02. Graduate School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, December 1994.

5. Bickmore, Timothy W. Aerojet Propulsion Division, Sacramento, California. Electronic correspondence, November 1994.

6. Bickmore, Timothy W. *Real-Time Sensor Data Validation*. NASA Contractor Report 195295. Contract NAS3-25883. Sacramento, California: Aerojet Propulsion Division, April 1994.

7. Bickmore, Timothy W. *SSME HPOTP Post-Test Diagnostic System Enhancement Project*: Final Report. Contract NAS3-25883. Sacramento, California: Aerojet Propulsion Division, 2 February 1994.

8. Boose, John H. "A survey of knowledge acquisition techniques and tools." London: Academic Press Limited, 1989.

9. Boose, John H. and Jeffrey M. Bradshaw. "Expertise transfer and complex problems: using AQUINAS as a knowledge-acquisition workbench for knowledge-based systems." London: Academic Press, Inc., 1987.

10. Buchanan, Bruce G. and David C. Wilkins. "Overview of Knowledge Acquisition and Learning," in *Readings in Knowledge Acquisition and Learning: Automating the Construction and Improvement of Expert Systems.* Eds. Bruce G. Buchanan and David C. Wilkins. San Mateo, California: Morgan Kaufmann Publishers, 1993.

11. Cendrowska, Jadzia. "PRISM: An algorithm for inducing modular rules," *International Journal of Man-Machine Studies, 27*: 349-370, 1987.

12. Cooper, Gregory F. "The Computational Complexity of Probabilistic Inference Using Bayesian Belief Networks." *Artificial Intelligence, 42*: 393-405, 1990.

13. Cosgrove, S. J. *Techniques for Knowledge Acquisition.* Research Note ERL-0553-RN. Salisbury, South Australia: Electronics Research Laboratory, Information Technology Division, June 1991.

14. Davis, Randall. "Interactive Transfer of Expertise: Acquisition of New Inference Rules," *Artificial Intelligence, 12*: 121-157 (1979).

15. Dubois, Didier, Henri Prade, Lluis Godo and Ramon López de Màntaras. "A Symbolic Approach to Reasoning with Linguistic Quantifiers," *Proceedings of the Eighth Conference on Uncertainty in Artificial Intelligence*, 74-82. San Mateo, California: Morgan Kaufmann Publishers, 1992.

16. Elsaesser, Christopher and Max Henrion. Verbal expressions for probability updates: How much more probable is 'much more probable'?" in *Uncertainty in Artificial Intelligence 5.* Eds. Max Henrion, Ross D. Shachter, L. N. Kanal and J. F. Lemmer. Amsterdam: Elsevier Science Publishers B.V. (North-Holland), 1990.

17. Eshelman, Larry, Damien Ehret, John McDermott and Ming Tan. "MOLE: a tenacious knowledge-acquisition tool." London: Academic Press, Inc., 1987.

18. Feigenbaum, Edward, A. *Knowledge Engineering: The Applied Side of Artificial Intelligence.* Palo Alto, California: Edward A. Feigenbaum, 1980.

19. Forsythe, Diana E. "Engineering Knowledge: The Construction of Knowledge in Artificial Intelligence," *Social Studies of Science, 23*: 445-477 (1993).

20. Forsythe, Diana E. and Bruce G. Buchanan. "Knowledge Acquisition for Expert Systems: Some Pitfalls and Suggestions," *IEEE Transactions on Systems, Man, and Cybernetics, 19*: 435-442 (May/June 1989).

21. Garey, Michael R. and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* San Francisco, California: W. H. Freeman and Co., 1979.

22. Geiger, Dan and David Heckerman. "Advances in Probabilistic Reasoning," *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence*, 118-126. San Mateo, California: Morgan Kaufmann Publishers, 1991.

23. Gilio, Angelo. "Conditional Events and Subjective Probability in Management of Uncertainty," in *Uncertainty in Intelligent Systems.* Eds. Bernadette Bouchon-Meunier *et alii.* Amsterdam: Elsevier Science Publishers B.V., 1993.

24. Goldszmidt, Moisés and Judea Pearl. "Reasoning With Qualitative Probabilities Can Be Tractable," *Proceedings of the Eighth Conference on Uncertainty in Artificial Intelligence*, 112-120. San Mateo, California: Morgan Kaufmann Publishers, 1992.

25. Gruber, Thomas R. "Automated Knowledge Acquisition for Strategic Knowledge," *Machine Learning, 4*: 293-336 (1989).

26. Hartrum, Thomas C. Lecture notes, Computer Science 594, Software Analysis and Design. Graduate School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, Fall 1994.

27. Heatherton, James R. *A Comparison of Knowledge Acquisition Techniques Used in the Development of Expert Systems.* M.S. thesis, AFIT/GLM/LSR/90S-23. Graduate School of Systems and Logistics, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, September 1990.

28. Heatherton, James R and Todd T. Vikan. *An Introduction to Expert Systems and Knowledge Acquisition.* Technical Report AFIT/LS/TR-90-1. Graduate School of Systems and Logistics, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, September 1990.

29. Henschen, L. "Inference," in *Encyclopædia of Artificial Intelligence.* Eds. Stuart C. Shapiro, David Eckroth and George A. Vallasi. New York: John Wiley & Sons, 1987.

30. Hillier, Frederick S. and Gerald J. Lieberman. *Introduction to Operations Research.* Oakland, California: Holden-Day, Inc. 1986.

31. Hink, Robert F. and David L. Woods. "How Humans Process Uncertain Knowledge: An Introduction for Knowledge Engineers," *AI Magazine*: 41-53, (Fall 1987).

32. Jackson, Bradley W. and Dmitri Thoro. *Applied Combinatorics with Problem Solving*. Reading, Massachusetts: Addison-Wesley Publishing Company, 1990.

33. Keshavan, H. R., Jeff Barnett, Dan Geiger and Thomas Verma. "Introduction to the Special Section on Probabilistic Reasoning," *IEEE Transactions on Pattern Analysis and Machine Intelligence, 15*: 193-195 (March 1993).

34. Kyburg, Henry E. Jr. "Subjective Probability: Criticisms, Reflections and Problems," in *Epistemology and Inference*. Minneapolis, Minnesota: University of Minnesota Press, 1893.

35. Loiseau, Stéphane. "A method for checking and restoring the consistency of knowledge bases," *International Journal of Human-Computer Studies, 40*: 425-442 (September 1994).

36. Marcus, Sandra and John McDermott. "SALT: A Knowledge Acquisition Language for Propose-and-Revise Systems." *Artificial Intelligence, 39*: 1-37 (1989).

37. Musen, Mark A. *Automated Generation of Model-Based Knowledge-Acquisition Tools*. London: Pitman Publishing, 1989.

38. Musen, Mark A. "Automated Support for Building and Extending Expert Models," *Machine Learning, 4*: 347-375 (1989).

39. Neapolitan, R. E. *Probabilistic Reasoning in Expert Systems: Theory and Algorithms*. New York: John Wiley and Sons, 1990.

40. Nutter, J. T. "Default Reasoning," in *Encyclopædia of Artificial Intelligence*. Eds. Stuart C. Shapiro, David Eckroth and George A. Vallasi. New York: John Wiley & Sons, 1987.

41. Pearl, Judea. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference* (Revised Second Printing). San Mateo, California: Morgan Kaufmann Publishers, 1991.

42. Reid, Thomas F. *Maintenance in Probabilistic Knowledge-Based Systems*. M.S. thesis, AFIT/GOR/ENS/86D-16. Graduate School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, December 1987.

43. *Reusable Rocket Engine Turbopump Health Management System.* Final Report. Contract NAS3-25882. San Diego, California: Science Applications International Corporation, 7 June 1994.

44. Rosenberg, Jerry M. *Dictionary of Artificial Intelligence and Robotics.* New York: John Wiley & Sons, 1986.

45. Ross, Sheldon. *A First Course in Probability.* New York: Macmillan Publishing Company, 1988.

46. Rumbaugh, James, Michael Blaha, William Premerlani, Frederick Eddy and William Lorensen. *Object-Oriented Modeling and Design.* Englewood Cliffs, NJ: Prentice Hall, 1991.

47. Sandahl, Kristian. "Transferring knowledge from active expert to end-user environment." *Knowledge Acquisition, 6 (1)*: 1-22, 1994.

48. Sander, Erik, David Foust, Mark Neely, Darryl Gaddy, Brian Piekarsky, Taylor Hooper and Randy Hurt. Marshall Space Flight Center, Huntsville, AL. Knowledge Acquisition Interviews conducted by June Zakrajsek, Claudia Meyer, T. W. Bickmore, Chris Fulton, Bill Maul and Catherine McLeod. NASA Lewis Research Center, Cleveland, Ohio. 29 April - 1 May 1992.

49. Santos, Eugene, Jr. "Computing with Bayesian Multi-Networks." Unpublished Report No. AFIT/EN/TR93-10. Air Force Institute of Technology, Wright-Patterson AFB, Ohio, 16 November 1993.

50. Santos, Eugene, Jr. "A Linear Constraint Satisfaction Approach to Cost-Based Abduction." *Artificial Intelligence, 65*: 1-28, 1994.

51. Santos, Eugene, Jr. "Efficient Jumpstarting of Hill-Climbing Search for the Most Probable Explanation," *Proceedings of International Congress on Computer Systems and Applied Mathematics Workshop on Constraint Processing*, 183-194, 1993.

52. Santos, Eugene, Jr. "A Fast Hill-Climbing Approach Without an Energy Function for Finding MPE," *Proceedings of the 5th IEEE International Conference on Tools with Artificial Intelligence*, 1993.

53. Santos, Eugene, Jr. A Fully Integrated Probabilistic Framework for Expert Systems Development. Proposal pursuant to Air Force Office of Scientific Research Grant #94-0006, 22 October 1993.

54. Santos, Eugene, Jr. "On the Generation of Alternative Explanations with Implications for Belief Revision," *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 337-347, 1991.

55. Santos, Eugene Jr. "Unifying Time and Uncertainty for Diagnosis." To appear in *Journal of Experimental and Theoretical Artificial Intelligence*, 1995.

56. Santos, Eugene Jr., Solomon Eyal Shimony and Edward Williams. "A Distributed Anytime Architecture with Co-operative Task Management for Probabilistic Reasoning," submitted to the *International Joint Conference on Artificial Intelligence*, 1995.

57. Scott, A. Carlisle, Jan E. Clayton and Elizabeth L. Gibson. *A Practical Guide to Knowledge Acquisition.* Reading, Massachusetts: Addison-Wesley Publishing Company, 1991.

58. Shachter, Ross D. and David E. Heckerman. "Thinking Backward for Knowledge Acquisition," *AI Magazine*, 55-61 (Fall 1987).

59. Shafer, Glenn. "Probability Judgement in Artificial Intelligence and Expert Systems," *Statistical Science, 2*: 3-44 (1987).

60. Shimony, Solomon Eyal. "Finding MAPs for belief networks is NP-hard." *Artificial Intelligence, 68*: 399-410, 1994.

61. Shoham, Y. and D. V. McDermott. "Temporal Reasoning," in *Encyclopædia of Artificial Intelligence.* Eds. Stuart C. Shapiro, David Eckroth and George A. Vallasi. New York: John Wiley & Sons, 1987.

62. Shortliffe, E. H. and B. G. Buchanan. "A model of inexact reasoning in medicine." *Mathematical Biosciences, 23*: 351-379, 1975.

63. Smith, Raoul. *The Facts on File Dictionary of Artificial Intelligence.* New York: Facts on File, 1989.

64. *Space Transportation System Training Data, SSME Orientation (Part A -- Engine).* Contract NAS8-40000. Course No. ME-110(A)RIR. Hyannis, Massachusetts: Rocketdyne Division, Rockwell International Corporation, 1 August 1988.

65. Spiegelhalter, David J., Rodney C. G. Franklin and Kate Bull. "Assessment, criticism and improvement of imprecise subjective probabilities for a medical expert system" in *Uncertainty in Artificial Intelligence 5*. Eds. Max Henrion, Ross D. Shachter, L. N. Kanal and J. F. Lemmer. Amsterdam: Elsevier Science Publishers B.V. (North-Holland), 1990.

66. Spivey, J. M. *The Z Notation: A Reference Manual.* Hertfordshire, England: Prentice-Hall International (UK) Ltd., 1989.

67. Srinivas, Sampath, Stuart Russell and Alice Agogino. "Automated construction of sparse Bayesian networks from unstructured probabilistic models and domain information in *Uncertainty in Artificial Intelligence 5*. Eds. Max Henrion, Ross D. Shachter, L. N. Kanal and J. F. Lemmer. Amsterdam: Elsevier Science Publishers B.V. (North-Holland), 1990.

68. Suter, Bruce. Lecture notes, Computer Science 531, Discrete Mathematics. Graduate School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, Fall 1994.

69. Tecuci, Gheorghe D. "Automating Knowledge Acquisition as Extending, Updating, and Improving a Knowledge Base," *IEEE Transactions on Systems, Man, and Cybernetics, 22*: 1444-1460 (November/December 1992).

70. Vikan, Todd T. *An Empirical Evaluation of Three Knowledge Acquisition Techniques for Developing a Project Management Related Expert System.* M.S. thesis, AFIT/GLM/LSR/90S-32. Graduate School of Systems and Logistics, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, September 1990.

71. Wilson, Robin J. *Introduction to Graph Theory*, (Third Edition). Essex, England: Longman Scientific and Technical, 1985.

72. Winston, Wayne L. *Introduction to Mathematical Programming: Applications and Algorithms*. Boston, Massachusetts: PWS-Kent Publishing, 1991.

73. Zakrajsek, June, Claudia Meyer, Chris Fulton and Bill Maul. NASA Lewis Research Center, Cleveland, Ohio. Personal interviews and electronic correspondence, June 1994 - February 1995.

Vita

Captain Banks was born on 29 April 1966 in Columbus, Ohio where he lived until graduation from the Columbus Academy in 1983. Subsequently, he studied Science and Technology in International Affairs at Georgetown University's Edmund A. Walsh School of Foreign Service in Washington, D.C., from which he was graduated in 1987. Upon receipt of his commission as a Second Lieutenant in the U.S. Air Force, Captain Banks attended Undergraduate Space Training at Lowry AFB, Colorado after which he was assigned to Cavalier AFS, North Dakota as a Missile Warning Operations Crew Commander and as the Chief of Standardization/Evaluation. In 1989 he received orders to Onizuka AFB, California where he served as the Officer-in-Charge of the Air Force Satellite Control Network's Command Post and as the Chief, Total Quality. He arrived at the Air Force Institute of Technology in May 1993.

Permanent Address: 3308 Penfield Road
Columbus, Ohio 43227-3744

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	March 1995	Master's Thesis

4. TITLE AND SUBTITLE
Acquiring Consistent Knowledge for Bayesian Forests

5. FUNDING NUMBERS

6. AUTHOR(S)
Darwyn O. Banks

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Air Force Institute of Technology
2950 P Street
WPAFB OH 45433-6583

8. PERFORMING ORGANIZATION REPORT NUMBER
AFIT/GSO/ENG/95M-01

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
June Zakrajsek
NASA Lewis Research Center/SPTD
21000 Brookpark Rd.
Cleveland, OH 44135
Mail Stop SPTD-2

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Distribution Unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This thesis develops a methodology and a tool for knowledge acquisition with the new probabilistic knowledge representation–the Bayesian Forest. It establishes the structure of the Knowledge Acquisition and Maintenance module of the Probabilities, Expert Systems, Knowledge and Inference (PESKI) architecture. The tool, MACK, is designed to be used directly by the domain expert(s) rather than by knowledge engineer(s), and thus supports automated knowledge acquisition.
This research determines and implements the constraints necessary to ensure the consistency of Bayesian Forest knowledge bases as data is both acquired and subsequently maintained. The impact to the PESKI architecture of time-dependent information and default assumptions during reasoning is also explored. The tool has been applied to NASA's Post-Test Diagnostics System which locates anomalies aboard the Space Shuttles' Main Engines.

14. SUBJECT TERMS
Automated knowledge acquisition, probabilistic reasoning, Bayesian Forests, Bayesian networks, expert systems

15. NUMBER OF PAGES
100

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18
298-102